DEVELOPMENT OF CONTROLLED COMPUTATIONAL EXPERIMENTS ON INTEGER LINEAR PROGRAMMING PROCEDURES

A THESIS

Presented to

The Faculty of the Division of

Graduate Studies

By

Benjamin Wei-Yuh Lin

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

in the School of Industrial and Systems Engineering

Georgia Institute of Technology

October, 1975

DEVELOPMENT OF CONTROLLED COMPUTATIONAL EXPERIMENTS

ON INTEGER LINEAR PROGRAMMING PROCEDURES

Approved:

R. L. Rardin, Chairman

R. G. Heikes

D. C. Montgomery

C. M. Shetty

W. R. Smyth

Date approved by Chairman: 10/31/75

ACKNOWLEDGMENTS

First and foremost, I would like to express my gratitude to Dr. Ronald L. Rardin, my advisor and friend. He was a constant source of inspiration, advice, and guidance throughout this dissertation research. I would also like to express my appreciation to the other members of my dissertation committee, Drs. Heikes, Montgomery, Shetty, and Smythe, for their numerous helpful suggestions on the content and scope of this study. Guidance and support provided by Dr. Lehrer and Dr. Hines is also greatly appreciated.

In addition to these faculty members, several fellow graduate students of mine were of great help to me in this research, especially Frank Alt and Bruce Schmeiser.

Finally, I wish to thank my wife Josefina for her patience and understanding during my graduate studies.

TABLE OF CONTENTS

Page

ACKNOWLEDGMENTS ii

LIST OF TABLES vi

LIST OF ILLUSTRATIONS vii

SUMMARY. viii

Chapter

I. INTRODUCTION 1

1.1 Purpose of the Research 2
1.2 General Concept of Controlled Computational Experiments 2
1.3 Scope and Limitations. 4
1.4 Plan of the Dissertation. 6

II. LITERATURE SURVEY. 8

2.1 General Integer Linear Programming Problem Formulation 8
2.2 Literature on Computational Complexity of ILP Problem 10

2.2.1 The Determinant of the Optimal Linear Programming Basis. 11
2.2.2 Number of Integer Variables 15
2.2.3 Parameters Related to Linear Programming Complexity 15

2.3 Literature on How ILP Procedures Are Compared. . . 17

2.3.1 How ILP Test Problems Are Obtained 17
2.3.2 How ILP Test Results Are Analyzed 19

III. PARAMETRIC GENERATION OF TEST PROBLEMS. 22

3.1 Rationale. 25
3.2 The Generating Procedure. 30
3.3 An Example 33

Chapter		Page
IV.	DEVELOPMENT OF AN EXPERIMENTAL APPROACH	38
	4.1 Selection of Response Variables	39
	4.1.1 Selection in Category 1	42
	4.1.2 Selection in Category 2	45
	4.1.3 Summary and Remarks.	46
	4.2 Treatment of Censored Data.	47
	4.2.1 Sampford and Taylor's Method for Randomized Block Designs	50
	4.2.2 The Procedure for Full Factorial Designs. .	53
	4.3 Application of the Analysis of Variance	56
	4.3.1 Verification of ANOVA Assumptions	57
	4.3.2 ANOVA with Censored Observations Estimated .	63
	4.4 Selection of Experimental Design.	64
V.	EMPIRICAL STUDY	75
	5.1 Outline of Test Algorithms Used	75
	5.2 Summary of the Experiment	78
	5.2.1 Selection of Low and High Parameter Values .	80
	5.2.2 Treatment of Time Limits	82
	5.2.3 Response Statistics Measured.	83
	5.2.4 Replication	83
	5.3 Analysis of Experimental Data.	83
	5.3.1 Choice of a Response Variable	83
	5.3.2 Occurrence of Censoring	85
	5.3.3 Preliminary Analysis of Data Transformations	86
	5.3.4 Estimation of Censored Values	90
	5.3.5 Final Analysis of Data Transformations . .	91
	5.4 Final Analysis of Variance and Empirical Results .	98
	5.4.1 Main Effects and Interactions for BB . . .	99
	5.4.2 Main Effects and Interactions for MIF. . .	106
	5.4.3 Summary.	109
VI.	OUTLINE OF CONTROLLED COMPUTATIONAL EXPERIMENTS . . .	111
	6.1 General Concept of Controlled Computational Experiments.	112

Chapter Page

6.2 Selection of Response Variables 113
6.3 Generation of Test Problems 113
6.4 Selection of Experimental Design. 115
6.5 Selection of Nuisance Parameters to Be Varied . . 116

6.5.1 Empirical Determination of Significant Nuisance Parameters. 116
6.5.2 Direct Use of Our Results on Nuisance Parameters. 117

6.6 Treatment of Censored Data and ANOVA with Censored Observations Estimated 118
6.7 Data Transformation 119

6.7.1 Empirical Determination of Data Transformation 119
6.7.2 Direct Use of Our Results on Data Transformations 120

VII. CONCLUSIONS AND RECOMMENDATIONS 122

APPENDIX

A. THE PROBLEM GENERATION CODE 127

B. REPLICATE I OF DATA COLLECTED. 140

C. REPLICATE II OF DATA COLLECTED 147

D. ANALYSIS OF VARIANCE TABLE FOR THE CUTTING PLANE RESULTS. 153

E. ANALYSIS OF VARIANCE TABLE FOR BRANCH AND BOUND RESULTS . 159

BIBLIOGRAPHY 165

VITA. 170

LIST OF TABLES

Table		Page
1.	Selection of Response Variables	47
2.	Basic Experimental Arrangement for a 2 by 2 Example	66
3.	Experimental Layout for Case I: Blocking on Ramdom Seeds.	68
4.	Expected Mean Squares for Case I: Blocking on Ramdom Seeds.	68
5.	Experimental Layout for Case II: Randomized Design	71
6.	Expected Mean Squares for Case II: Randomized Design.	71
7.	Accuracy Tolerances Used in Computer Codes	79
8.	Problem or Nuisance Parameters Varied in Experiments	81
9.	High and Low Level of Parameters Used in Experiments	82
10.	Correlation Coefficients of Solution Time vs. Equivalent Iterations from LP to IP Optimality and Solution Time vs. LP Subproblems.	84
11.	Occurrence of Censored Observations.	85
12.	Sum of Squares for BB	97
13.	Sum of Squares for MIF	97
14.	Analysis of Variance Table for Branch and Bound Results	100
15.	Analysis of Variance Table for Cutting Plane Results	101
16.	Estimates of Main Effects for BB.	102
17.	Estimates of Main Effects for MIF	102
18.	Selection of Response Variables	114
19.	Nuisance Problem Parameters to be Varied	117
20.	Transformation Used in Empirical Study.	120

LIST OF ILLUSTRATIONS

Figure		Page
1.	A Framework for Comparison of ILP Procedures	3
2.	An Illustration of Face Generation	13
3.	Scatter Diagram of Mean vs. Variance (Untransformed BB Observations)	87
4.	Scatter Diagram of Mean vs. Variance (Untransformed MIF Observations)	87
5.	$L_{max}(\lambda)$ Against λ for BB (Prior to Application of Censoring Procedure)	89
6.	$L_{max}(\lambda)$ Against λ for MIF (Prior to Application of Censoring Procedure)	89
7.	$L_{max}(\lambda)$ Against λ for BB (Censoring Procedure Used). . .	92
8.	$L_{max}(\lambda)$ Against λ for MIF (Censoring Procedure Used) . .	92
9.	Histogram of Error Residuals for BB (Transformed Observations with Estimated Censor Points)	93
10.	Histogram of Error Residuals for MIF (Transformed Observations with Estimated Censor Points)	93
11.	Scatter Diagram for Transformed Branch-and-Bound Observations with Estimated Censor Points.	94
12.	Scatter Diagram for Transformed Cutting Plane Observations with Estimated Censor Points.	95
13.	Graph of Number of Variables vs. Number of Constraints Interaction in Branch and Bound Results.	105
14.	Graph of Distance Index vs. Number of Constraints Interaction in Cutting Plane Results.	108
15.	A Framework for Comparison of ILP Procedures	112

SUMMARY

Testing and comparison of mathematical programming algorithms is an integral part of the algorithm development process. This dissertation attempts to develop a framework for comparing all-integer linear programming (ILP) procedures. Several ILP problem characteristics are suspected to greatly affect algorithm performance and thus become "nuisance parameters" to the algorithm developers. Examples are the number of integer variables, the number of constraints, the determinant of the optimal linear programming basis, and the density of nonzero entries in the constraints.

An ideal framework for comparing algorithms is to be able to control the settings of these nuisance factors and at least partially eliminate their effects. An outline of conducting a computational study of ILP procedures in such a framework is developed in this dissertation research. Various design issues in the development of the experimental approach are investigated, and a random test problem generator is presented which controls many of the nuisance parameters other researchers have indicated might affect the difficulty of ILP problems.

A major empirical study of the effect of various nuisance parameters on solution times for cutting plane and branch-and-bound algorithms is included in the dissertation. This study both provides empirical insights on the experimental design issues developed in the dissertation, and provides empirical evidence on the question "What makes integer programs hard to solve?" Many conjectures in the

literature are empirically confirmed for the first time, and some new effects are identified.

CHAPTER I

INTRODUCTION

Integer linear programming problems (ILP's) are constrained optimization problems in which the objective function and constraints are linear and some of the decision variables are required to have integer values. This type of problem abounds in various branches of engineering, business, and the physical and social sciences.

During the past fifteen years, numerous solution procedures have been advanced for ILP problems. Often several different solution strategies have been proposed for the same type of ILP problem. A very important problem to be posed is: How is the efficacy of one solution procedure to be compared to that of other existing procedures or procedures under development? While this problem may be of equal importance to the problem of how to develop solution procedures, little work has been reported in the literature on techniques for comparison of solution procedures. Thus each author is "on his own" in reporting experience with his proposed algorithm.

Typically a small number of randomly generated problems are run by both the proposed procedure and some alternatives and results are compared in terms of solution time, number of simplex iterations, etc. In a strict sense any conclusions drawn from such computational analyses are valid only for the given problems. However, if test problems are randomly generated--so that solution times are observations on random variables--much stronger inferences are possible if proper attention is

paid to experimental design. It is the need for such attention to experimental design that motivates the work of this dissertation.

1.1 Purpose of the Research

The specific purpose of this dissertation research is to develop an approach to controlled computational experiments on integer linear programming procedures. It is assumed throughout that test problems for such experiments are to be randomly generated so that techniques can be applied which take full advantage of the power of statistical experimental design. It is hoped that the results of this research will provide a more powerful means of evaluating and comparing integer linear programming procedures than is presently available. By investigating various design issues encountered in the experimental studies of integer linear programming procedures, insight into conducting of better experiments should be realized.

1.2 General Concept of Controlled Computational Experiments

Controlled experiments are experiments in which nuisance factors not of main concern (but affecting the response) are controlled through experimental design. The advantage of such experiments is improved information about the factors of interest by reducing the effects of nuisance factors. This dissertation is concerned with comparison of ILP solution procedures; however, several nuisance factors are suspected to greatly affect algorithm performance. Examples are the number of integer variables (denoted n), the number of constraints (denoted m), the density of nonzero entries in the constraints, etc.

An ideal framework for comparing solution procedures is to be

able to control the settings of these nuisance factors and at least partially eliminate their effects. Such a framework is illustrated in Figure 1. By experimenting with all algorithms at all levels of the nuisance parameters, nuisance effects can be eliminated and true differences between algorithms measured.

Nuisance Factors / Algorithms	Density (H)				Density (L)			
	m(H)		m(L)		m(H)		m(L)	
	n(H)	n(L)	n(H)	n(L)	n(H)	n(L)	n(H)	n(L)
Algorithm 1								
Algorithm 2								
.								
.								
.								
Algorithm P								

H = High Level and L = Low Level

Figure 1. A Framework for Comparison of ILP Procedures

To enable one to do a computational study of ILP procedures in such a framework, a number of tasks need to be undertaken:

1. Thoroughly identify nuisance factors and develop a random test problem generator which can control these nuisance factors to a maximum degree.

2. Reduce the cost and time of experimentation by determining which nuisance factors significantly affect results. Only significant nuisance factors need to be taken into consideration.

3. Specialize the techniques of statistical experimental design to deal with the peculiar problems of computational experiments in the algorithms versus nuisance parameter framework.

Completion of these tasks is the objective of this dissertation research.

1.3 Scope and Limitations

While any ILP experiment on randomly generated problems can be approached in the algorithm versus nuisance parameter format of Figure 1, the generality of research results in this dissertation is limited in a number of ways. Among these are that:

1. Analysis is limited to pure ILP problems and to linear programming-based algorithms for such problems. 0-1 ILP and mixed-integer problems are not addressed.*

2. Empirical studies are performed on only two generic ILP solution procedures which use linear programming to solve subproblems, namely Gomory's fractional dual cutting plane algorithm and a Land-and-Doig type of enumerative algorithm. These algorithms were chosen as representative of the cutting plane and branch-and-bound classes of linear programming-based ILP algorithms, but empirical results are only strictly valid for these algorithms.

3. All test problems used in empirical studies were generated with the random problem generator developed in Chapter III. The generator is designed to randomize all elements of problem structure

---

*Throughout this dissertation "Pure ILP problems" is meant to exclude 0-1 ILP problems; and "LP-based ILP algorithms" means that the linear programming relaxation of each candidate ILP subproblem is solved by the simplex method.

not completely controlled, but other generators might produce different results.

The reasons for limiting the analysis of this dissertation to pure ILP problems and LP-based ILP algorithms are two-fold. First it seems reasonable to believe that the nuisance or problem parameters for pure ILP problems would differ from those of 0-1 ILP or mixed-integer problems. For example, Balas (1) noted the underlying group structure of 0-1 problems is different from the one for general problems. Similarly, different parameters would probably need to be used for LP-based and for non-LP based algorithms because parameters which complicate LP aspects of an LP-based procedure would have no effect on non-LP procedures. The other reason for the limitation to LP-based procedures is a widely held conclusion--reached for example by Geoffrion and Marsten (19)--that all of the successful general purpose ILP codes are LP-based. Thus LP-based algorithms are the ones of greatest practical interest to researchers.

Despite these formal limitations it is believed that much of the development of this dissertation is applicable to the full spectrum of ILP algorithms. Only some aspects of the random problem generator developed in Chapter III, the response variable discussion in Section 4.1, and some of the empirical results of Chapter V appear to be actually limited to the pure-integer-LP-based case.

The results produced by empirical studies in this dissertation may have significance in many other contexts than experimental design. The analysis of the significance of nuisance factors necessary for development of controlled computational experiments can also be viewed

as empirical investigation of the question "What problem characteristics make integer programs hard to solve?" Insights gained on this question have value in a wide range of integer programming research. For example, they might suggest which of several formulations of a given problem would be easiest to solve.

1.4 Plan of the Dissertation

The chapters of the dissertation which follow this brief introduction attempt to provide an approach to controlled computational experiments on integer linear programming procedures. In Chapter II a brief review is performed of the current literature on how ILP solution procedures are compared and what parameters make ILP problems difficult to solve.

In Chapter III, a parametric generator of ILP test problems is developed which can control the parameters identified in Chapter II to a maximum degree. An illustrative example of the generation process is given.

In Chapter IV, attention is directed to several important experimental design and data analysis issues arising in the conduct of empirical study of ILP solution procedures. Each issue is presented and discussed analytically.

Chapter V presents a detailed empirical study, using the results in Chapter IV. In addition to identifying parameters of ILP problems which significantly correlate with the computational complexity of ILP problems, this chapter seeks to provide experience with data analysis and experimental design of ILP solution procedures. Thus it adds empirical insights to the analysis of Chapter IV.

Chapter VI outlines a general approach to conducting controlled computational experiments on ILP solution procedures. Experience gained in this research is synthesized to yield a proposed procedure for experiments on ILP's.

In Chapter VII the dissertation concludes with a brief summary of important results and recommendations for future research.

CHAPTER II

LITERATURE SURVEY

As outlined in Chapter I, this dissertation is addressed to the development of appropriate designs for computational experiments on algorithms for integer linear programming (ILP) problems, where such experiments are controlled by testing algorithms at various levels of nuisance or problem parameters. In this chapter previously reported research which is particularly relevant to this purpose is surveyed. In particular, two categories of research are addressed:

1. Characteristics or parameters of ILP problems which appear relevant to the difficulty of the problems; and

2. Previous approaches to comparing ILP algorithms.

Any other literature which may be relevant to particular topics of the dissertation will be discussed in the contexts of these topics. All literature is discussed in terms of the ILP problem formulation given in the next section.

2.1 General Integer Linear Programming Problem Formulation

Consider the ILP

$$\begin{aligned} \text{Max } x_0 &= cx \\ Ax &= b \\ x &\geq 0, \text{ integers} \end{aligned} \qquad \text{(ILP)}$$

and the corresponding LP

$$\begin{aligned} \text{Max } x_o &= cx \\ Ax &= b \qquad \text{(LP)} \\ x &\geq 0 \end{aligned}$$

Let B be a basis matrix for (LP), and partition $x = (x_B, x_N)$, $A = (B,N)$ and $c = (c_B, c_N)$. In partitioned form (LP) is

$$\begin{aligned} \text{Max } x_o &= c_B x_B + c_N x_N \\ B\, x_B + N\, x_N &= b \\ x_B, x_N &\geq 0 \end{aligned}$$

Expressing x_o and x_B in terms of x_N yields

$$\begin{aligned} \text{Max } x_o &= c_B B^{-1} b - (c_B B^{-1} N - c_N) x_N \\ x_B + B^{-1} N x_N &= B^{-1} b \qquad \text{(LP1)} \\ x_B, x_N &\geq 0 \end{aligned}$$

If $B^{-1}b \geq 0$, B is called a primal feasible basis; if $c_B B^{-1} N - c_N \geq 0$, B is called a dual feasible basis. If B is primal and dual feasible, it is an optimal basis. Rewriting ILP in terms of an optimal basis B, we obtain the equivalent form.

$$\begin{aligned} \text{Max } x_o &= c_B B^{-1} b - (c_B B^{-1} N - c_N) x_N \\ B x_B + N x_N &= b \qquad \text{(ILP1)} \\ x_B, x_N &\geq 0, \text{ integer.} \end{aligned}$$

Research on solving ILP can be divided into three main streams. Historically, cutting plane algorithms were the first to be proposed. These methods generate new linear constraints so as to eventually obtain a linear program whose optimal solution is integral. The best possible cutting planes are faces of the convex hull of integer solution to ILP, but such cuts are generally very difficult to characterize.

The second stream of development, derived partly from the theory of cutting plane algorithms, concentrates on the characterization of faces of the convex hull of feasible integer solutions to an "asymptotic" problem derived from (ILP1) and associated with an underlying factor group. This development, largely advanced by Gomory, is usually called the group-theoretic approach.

The third stream of work, called the enumerative method, is concerned with procedures for intelligent enumeration over all possible solutions. Many (but not all) enumerative approaches use bounds derived from the solution of LP as the principal scheme for restricting the number of solutions explicitly enumerated.

Our concern in this dissertation is LP-based solution procedures for ILP's. It thus encompasses both the cutting plane and group theoretic approaches which are inherently linear programming oriented, together with that part of research on enumerative algorithms which deals with bounds obtained from solving LP.

2.2 Literature on Computational Complexity of ILP Problem

The fact that ILP problems are generally difficult to solve has been noted since Gomory gave a proof of the first finitely convergent

cutting plane algorithm. Though that algorithm was theoretically convergent, Thompson (53) reported that even for a small ILP problem, Gomory's algorithm might sometimes require generating a tremendous number of cuts to reach optimality.

Much later Jeroslow and Kortanek (32) constructed a class of two-variable integer programming problems which are indexed by a positive continuous parameter. They have shown that Gomory's fractional algorithm may require arbitrarily many cuts to be added to the dual simplex tableau before convergence to an optimum integer solution occurs for the class of problems. Thus, while there is a finite bound on the number of cuts before an integer solution for each problem of this type, there is no overall bound for all problems of this type.

In order to select nuisance parameters which should be controlled in computational experiments, it is important to understand what characteristics of ILP's bring about such difficulties. The next few sections outline the available literature on the problem parameters which appear to make ILP's difficult.

2.2.1 The Determinant of the Optimal Linear Programming Basis

If the absolute value of the determinant of B in (ILP1) is 1, it is obvious that the corresponding optimal solution for LP is integer and thus an optimal solution to ILP. For this reason many network-based ILP's are known to be relatively easy integer programs. The constraint matrix A for such problems can be shown to be unimodular, i.e., to have only basis matrices with absolute value of determinant $\pm$ 1.

Two groups of research in the integer programming literature suggest that the determinant of the optimal linear programming basis

may be a key element in the computational complexity of all ILP's. Jeroslow (30) has shown that using only two inequalities, there can be found an ILP problem such that the number of linear inequalities needed to define the convex hull of the integer points contained within the given two inequalities exceeds any given integer.

The idea of how a great number of faces can be constructed in the two dimension is briefly discussed. Figure 2 shows two constraints $a_1 \cdot x = b_1$ and $a_2 \cdot x = b_2$ intersecting at point P. $Q_1Q_2, Q_2Q_3, \ldots, Q_{N-1}Q_N$ are the facial segments forming the integer hull of the feasible points for the constraints. Consider the face having facial segment $Q_{N-1}Q_N$. Note that the triangle PQ_NS has no integral points in it other than those on $Q_{N-1}Q_N$. It is always possible to lift the line PQ_N in the direction indicated by the arrow such that the triangle PQ_NR contains no integral points, where PR is represented by $a_2'x = b_2$. If the triangle had admitted some integral point other than Q_N into PQ_NR, it must lie off the segment PQ_N. Any undesired point can be excluded by lowering the line PR, and the procedure is repeated as many times as is necessary. As PQ_NR can contain only finitely many integral points, such excluding procedure needs to be involved only a finite number of times. Many more faces can be constructed as such.

It seems obvious that number of faces is directly related to computational difficulty of cutting plane procedures. If many faces are required to define the convex hull of integer solutions, many cuts will likely be needed to obtain an integral solution for the LP relaxation of ILP problems.

Motivated by this observation, Jeroslow (31) further investigated

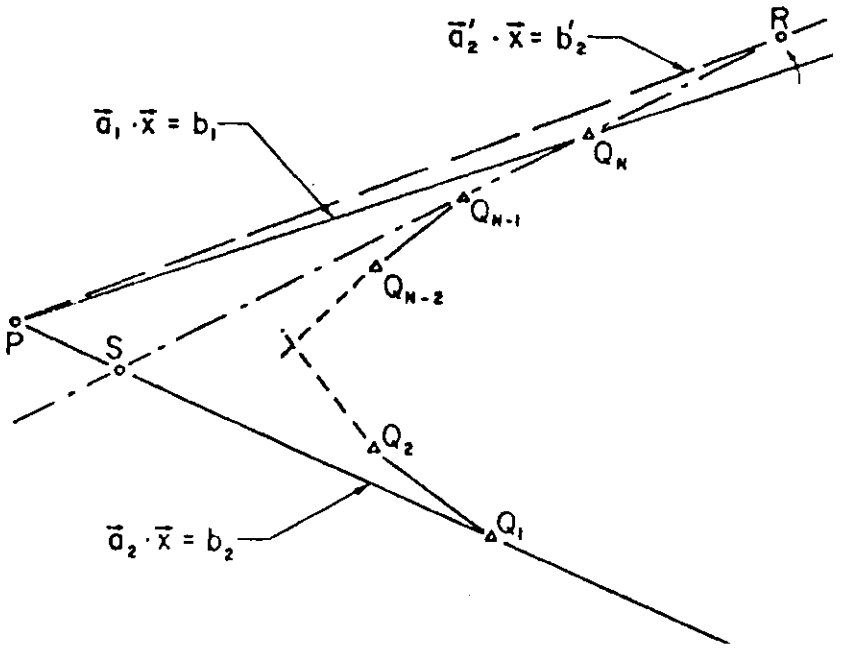

Figure 2. An Illustration of Face Generation
(Source: Jeroslow (30))

the maximum possible number of faces. He showed that the logarithm (to base 2) of the absolute value of the determinant is essentially an upper bound or the possible number of faces in the two-dimensional case. Jeroslow thus suggested that the determinant of the LP-optimal basis would probably be a relevant measure of the "hardness" of ILP's.

The group theoretic approach provides the other segment of integer programming research which indicates that the determinant of the LP-optimal basis may be a significant parameter in computational complexity of ILP's. Let ILP2 be the relaxation of ILP1 obtained when the constraint $x_B \geq 0$ is dropped. Gomory showed in (23) that ILP2 can be transformed into a knapsack-type problem over a finite Abelian

group. The decision variables in this problem are integer-valued functions t(g) of the group elements g which are members of a set η. This set is the set of group elements corresponding to the fractional parts of columns of $B^{-1}N$ in ILP1 under the mapping which produces the group problem. Thus the t(g) for η correspond directly to the values of x_N in ILP.

If the group problem has a feasible solution, Gomory showed that $\pi_{g\in\eta} (1 + t(g)) \leq D$, where D is the absolute value of the determinant of the LP-optimal basis. This leads to

$$\pi_j (1 + x_{Nj}) \leq D$$

where x_{Nj} is the jth component of x_N in ILP1.

When D = 1 (which is the case when the matrix A is unimodular), the above inequality shows that $x_{Nj} = 0$ for all j. The optimum integer solution is identical to the linear programming solution. As D increases from 1, however, the solution gradually moves away from the linear program solution to the general integer program solution. Nonbasic variables in ILP1 may take on larger and larger values. In any LP-based ILP algorithms such disparity from the LP solution could be expected to produce complications in solution of ILP problems.

The above results follow because the order of the group underlying Gomory's formulation is bounded by D (see (23) for a proof). Since the group may be as large as D, it follows that D also corresponds to the complexity of solving Gomory's group formulation directly. Knapsack procedures used for example in Gorry and Shapiro (24) have efficiency directly proportional to the order of the underlying group.

2.2.2 Number of Integer Variables

Perhaps the most widely mentioned measure of the complexity of integer linear programming problems is the number of integer decision variables in the problem. Presumably because the number of integer vectors which might provide a solution to a bounded integer programming problem in n-space grows exponentially in n, many researchers have routinely assumed ILP problems become more difficult as the number of integer variables increases.

Careful empirical investigations have generally tended to confirm the role of the number of integer variables in the computational complexity of ILP's. For example, Geoffrion (20) reports results for three classes of problems in the range of 30-90 variables. The problem classes were set covering, optimal routing in networks, and knapsack type with several constraints. With LP-based enumeration algorithms, ninety-variable problems of the first two varieties solved routinely in well under one minute, with solution times increasing approximately as the square of the number of variables. Eighty-variable problems of the third variety solved routinely in approximately four minutes, with solution times increasing approximately as the fourth power of the number of variables. Pierce (45) made similar observations.

2.2.3 Parameters Related to Linear Programming Complexity

As mentioned earlier, the scope of this research is limited to LP-based ILP solution procedures. Since such procedures involve repeatedly solving linear programs, any factors which make LP hard to solve are potential factors which make ILP's hard. Dantzig (13) estimated the total number of computer multiplications (or divisions) for

the revised simplex procedure as follows. Assume that an m x n system is in a feasible canonical form. The total number of multiplications required per iteration is

$$t(n-m)(m+1) + tm(m+1) + (m+1)^2 = tn(m+1) + (m+1)^2$$

where the fraction of nonzero entries in the original tableau and in the column entering in the basis are assumed on the average to be both equal to t. The three terms on the left are the number of multiplications used (a) in "pricing out," (b) in representing the new column, and (c) in pivoting.

The above estimate suggests that the number of constraints and density--in addition to the number of variables--are potential factors affecting the complexity of ILP problems. However, recent computational experience with ILP problems demonstrates that some of the guiding principles from LP can be dangerous. Anyone accustomed to working LP applications is inclined to economize on the number of constraints he uses in a large-scale model. Beale and Tomlin (4), Geoffrion and Graves (18), and Williams (59) have observed that a great reduction of solution time can be achieved by adding new constraints that are redundant in an ILP sense but are not so for the associated LP relaxation. The reason is that better bounds are likely to be obtained from the LP relaxation of such constraints are added. Direct computational confirmations of this observation is included in all the above studies. Whether more or fewer of constraints make ILP's difficult remains unanswered; however, the number of constraints surely is a factor.

2.3 Literature on How ILP Procedures Are Compared

During the past fifteen years, numerous solution procedures have been proposed for ILP problems. Several different solution strategies have often been proposed for the same type of ILP problem. Almost without exception, any report of the development of an ILP solution procedure contains some computational experience on the proposed one as compared to that of other existing procedures. The presentation which follows is to give a brief review of these computational comparisons of ILP algorithms.

2.3.1 How ILP Test Problems Are Obtained

Before the performance of ILP procedures can be evaluated, test problems must be obtained. The need for test problems has long been recognized. Problems for testing ILP algorithms have typically been either obtained from real-world situations, developed by particular authors to illustrate algorithmic behavior, or randomly generated.

Real-world problems would probably provide the most valid tests of algorithms if enough were available. However, very few large real-world problems have been documented in the literature. Kuehn and Hamburger (36) developed some real-world warehouse location problems for testing their heuristic program. Their models assume a factory-warehouse-market supply system, where the factory site is at either Indianapolis (problem set 1), Jacksonville (problem set 2), or at both Indianapolis and Baltimore (problem set 3). In each of the problem sets there are 24 potential warehouse sites, among 50 market places. Data and locations of market and warehouse sites are detailed

in the study. Due to the completeness of model description, a number of studies (such as Khumawala (34), McGinnis (40), and Sâ (48)) specializing in warehouse location problems have used the Kuehn and Hamburger problems for testing the performance of their procedures.

The second source of test problems comes from problems which are designed by authors to illustrate algorithmic behavior. The most widely used of such test problems are probably the ones compiled by Haldi (27). These test problems fall into three separate groups. The first set consists of ten small fixed charge problems developed by Haldi (26) which characteristically contain a number of zero-one variables, each of which is identified with a "lump-sum" fixed charge. The second set consists of six problems of the machine-scheduling type developed by Giglio and Wagner (21). All of these problems have similar dimensions, i.e., six jobs are required to follow the same processing sequence on three separate machines. The third set is usually called "IBM" test problems provided by IBM. The origin of IBM problems was not known nor is it clear whether they illustrate any particular type of application.

Haldi (28), Trauth and Woolsey (54), Trotter and Shetty (55), among others, have used the Haldi problems to evaluate the performance of their ILP procedures. Unfortunately, the Haldi problems are rather small for making meaningful comparisons of current day ILP algorithms. More recently, Wahi and Bradley (58) have compiled another set of test problems developed by others and given relevant information about solutions to these problems.

The third and last source of test problems is random generation. Many studies use randomly generated problems, but no standard

generation scheme has evolved. For example, Nemhauser and Ullman (44) randomly generated knapsack test problems in which the largest problem conformed to the following rules: the cost and constraint coefficients were uniformly random between 0 and 99; the right side b was set at 2500; and each of the 50 integer variables is upper bounded by 2.

Since randomly generated problems typically provide the only practical way to obtain sufficient numbers of large test problems for any mathematical programming algorithm, a substantial literature of random problem generators has recently developed (see (10), (35), and (41)). The availability of a network generator seems to have made possible an extensive computational study on transportation problems by Glover, Karney, Klingman and Napier (22). Unfortunately, no such systematic scheme has appeared in the ILP literature.

2.3.2 How ILP Test Results Are Analyzed

A great number of ILP studies attempt to both develop the theory of a proposed procedure and compare the algorithmic behavior of the procedure to that of other algorithms. Typically comparisons are made on the basis of differences in average solution time, average number of simplex iterations, etc. for some small number of test problems developed as discussed in Section 2.3.1. Conclusions which can be drawn differ with the source of test problems.

A first category consists of the first two sources of test problems discussed in Section 2.3.1, namely real-world problems and problems designed by others. In a strict sense conclusions drawn from such comparisons are valid only for the given test problems. For example, Trauth and Woolsey (54) used Haldi's problems to test Gomory's method

of integer forms against his dual all-integer algorithms. A total of five codes based on the two algorithms were compared. For economic reasons, an arbitrary upper bound of 7,000 iterations was assigned as a cutoff point in the calculations. While the whole purpose of their paper was to report on computational efficiency of ILP algorithms, only vague general conclusions could be drawn.

It is rather surprising that very little attempt has been made to draw more precise conclusions from experiments on random problems. When test problems are randomly generated--and thus solution times are realizations of a random variable--strong statistical inferences can be made. Appropriate use of statistical analyses and experimental design can make conclusions drawn from particular test problems valid for all randomly generated problems of given classes.

One example of the typical treatment of results from random problems is Greenberg and Hegerich's (25) test of knapsack problem procedures. Average number of branch and bound nodes generated and average solution times in milliseconds were collected to compare the procedures. Ten problems were run to obtain each average. While conclusions were drawn, no attempt was made to assess the statistical significance of apparent differences in the algorithms.

Recently Rardin (46), and Rardin and Unger (47) made the first known attempt to draw such statistical inferences by applying the analysis of variance in a computational comparison of various fixed charge network algorithms. Randomly generated test problems were used which possessed all combinations of the properties previous researchers have indicated most affected computational efficiency of algorithms for

fixed charge problems. Rather strong statistical conclusions were drawn. For example, the authors used the significance level in the analysis of variance to determine to what degree various problem parameters affected solution times. However, many statistical issues were not resolved. In particular, no attempt was made to verify the statistical assumptions on which the analysis of variance is based. The remainder of this dissertation is an attempt to explore such statistical issues so that other researchers can take advantage of the analytical power of statistical inferences about solutions to randomly generated test problems.

CHAPTER III

PARAMETRIC GENERATION OF TEST PROBLEMS

The need for test problems has long been recognized in the field of mathematical programming. As mentioned in Section 2.2, test problems for ILP algorithms--and indeed for all mathematical programming algorithms--have typically been either obtained from real-world situations, or developed by particular authors to illustrate algorithmic behavior, or randomly generated. Real-world problems would probably provide the most valid tests of algorithms if enough were available. However, very few large real-world problems have been documented in the literature. Similarly, test problems designed by authors to illustrate algorithmic behavior are too small to provide useful evaluation of present-day ILP algorithms. Thus researchers desiring to compare the performance of their ILP algorithms to that of others proposed in the literature have been forced to turn to randomly generated problems.

Since randomly generated problems typically provide the only practical way to obtain sufficient numbers of large test problems, a substantial literature of random problem generators has recently developed. In particular several authors have reported parametric problem generators which produce test problems having particular properties desired by experimenters. Such an LP generator has been developed by Charnes, Raike, Stutz, and Walters (10), a network generator was reported by Klingman, Napier, Stutz (35), and a nonlinear programming generator was proposed by Michaels and O'Neill (41).

As noted by Michaels and O'Neill (41), besides providing the only currently practical method for testing programming codes, a parametric random problem generator has several distinct advantages:

1. The virtually limitless supply of test problems;
2. The ability to know and control problem characteristics;
3. The ability to conduct computational studies of the effects of parameter variation;
4. The option to regenerate at will rather than to store data for large test problems.

While several random problem generators have been used in testing ILP solution procedures, there has been no standardization--and little real discussion--of generator design. Some generators use the number of constraints and the number of integer variables as parameters. Some add the density of the constraint matrix to the above two parameters. Some use parameters specific to problems with special structure. In testing fixed charges problems, Spielberg (51) for example, uses the number of plants plus the number of customers as parameters.

In this chapter the design of an ILP problem generator is approached more systematically. A generator is presented which maintains control of those parameters that the literature of integer programming suggests have a significant effect on problem difficulty. Our major departure from other parametric ILP problem generators is based on an observation by Jeroslow (30). He advocates that the absolute value of the determinant of the LP-optimal basis should be used as a measure of complexity of ILP problems, rather than traditional measures such as the number of constraints and the number of integer variables, etc.

Thus, the development of a parametric problem generator in this chapter has been focused on how to control the LP-optimal basis to a maximum degree.

Since there is some inconsistency in reports of what parameters make ILP problems hard, other parameters which could seemingly be significant were also incorporated in the development of the problem generator. Specifically, the following items are controlled in the generator to at least some degree:

1. The density of nonzero entries in the constraint matrix
2. The number of constraints
3. The number of integer variables
4. The degree of primal degeneracy in the optimal solution of the LP relaxation
5. The degree of dual degeneracy in the optimal solution of the LP relaxation
6. An index of the distance between the optimal solutions to the LP relaxation and ILP.

In addition all generated problems are feasible and bounded and have integer costs and constraint matrices.

The first three of the above parameters have at least been suggested in the literature reviewed in Chapter II, and it seems obvious that the distance between LP and IP solutions would affect the efficiency of an LP-based ILP procedure. Primal and dual degeneracy are included because they are key elements in convergence proofs of the LP algorithms on which LP-based ILP algorithms rely, and because it can at least be argued that they affect the ILP directly. A solution

to an LP relaxation which is highly primal degenerate automatically has large number of variables taking on integral (zero) values in the LP optimal solution. Similarly a solution which is highly dual degenerate has many alternative optimal solutions to the LP relaxation, perhaps including an all-integer solution.

3.1 Rationale

Before stating precisely a procedure for generating ILP problems, we describe the rationale of the procedure and justify the key steps. Let the ILP problem be

$$\begin{aligned} &\text{Max } c_B x_B + c_N x_N \\ &\quad Bx_B + Nx_N = b \qquad \text{(ILP1)} \\ &\quad x_B, x_N \geq 0 \text{ and integer} \end{aligned}$$

To be specific, let B is a m x m matrix and N be a mx(n-m) matrix, where the elements of both matrices are all integral. x_B, c_B, and b are m-vectors, x_N and c_N are (n-m)-vectors; and c_B, c_N, and b are integer. Furthermore, let the determinant of B be D. We will show how the LP basis B is generated which is LP-optimal and has the desired determinant, how the non-basic matrix N is generated to obtain feasibility and the desired density of nonzero entries, and how the distance between LP optimality and ILP optimality is partially controlled.

Assume

$$BX_B = b \qquad (3\text{-}1)$$

It is well known that any integer matrix B can be transformed into the Smith diagonal matrix S, i.e.,

$$RBC = S \tag{3-2}$$

where R and C are unimodular matrices of elementary row and column operations, respectively. Recall that the Smith diagonal matrix is a diagonal matrix with diagonal elements $d_1, d_2, \ldots, d_m$ having the following properties:

1. $D = \pi_{i=1}^{m} d_i$ = determinant of S
2. $d_1, d_2, \ldots, d_m$ are integers
3. d_i divides d_{i+1} for $i = 1,2,\ldots,m-1$

Since R and C are unimodular, R^{-1} and C^{-1} exist and are also unimodular. Thus (3-2) leads to

$$B = R^{-1} S C^{-1} \tag{3-3}$$

Substituting (3-3) into (3-1) gives

$$R^{-1} S C^{-1} x_B = b \tag{3-4}$$

To generate the elements of (3-1), we can start by generating x_B in (3-4), generate C^{-1} and apply it on the left to x_B, generate S and apply it on the left to $C^{-1}x_B$, and generate R^{-1} and apply it on the left to $SC^{-1}x_B$. If S is generated in the Smith diagonal format and R^{-1} and C^{-1} are kept unimodular the absolute value of the determinant of B will obviously be D.

Let x_B be any nonnegative rational vector whose elements are of the form h_i/d_i for some random nonnegative integers h_i. The number of

zero h_i, i.e., the number of zero components in x_B, controls the degree of primal degeneracy. Since x_B is nonnegative and finite, it has the proper form for an LP optimal solution. Moreover, the existence of such a finite optimal solution to the LP1 relaxation assures ILP1 is bounded.

To maintain integrality of $b = R^{-1}SC^{-1}x_B$, two approaches can be used. One way is to randomly generate C^{-1} and obtain $SC^{-1}x_B$. Since $SC^{-1}x_B$ may not be an integral vector, R^{-1} must be randomly, but carefully, chosen to assure integrality of b. The other approach is to randomly generate a unimodular matrix C^{-1} such that $C^{-1}x_B$ is of the form:

$$C^{-1}x_B = \begin{pmatrix} g_1/d_1 \\ g_2/d_2 \\ \vdots \\ g_m/d_m \end{pmatrix} \qquad (3\text{-}5)$$

for some integers $g_1, g_2, \ldots, g_m$.

If $C^{-1}x_B$ satisfies (3-5), it is obvious that $S(C^{-1}x_B)$ is an integral vector. Thus R^{-1} can be any unimodular matrix. The generating procedure presented in the next section uses this latter approach with R constructed as the product of elementary operation matrices.

A C^{-1} can be generated which has the desired properties by taking C^{-1} in the form

$$C^{-1} = \begin{pmatrix} 1 & & & & 0 \\ \alpha_2 & 1 & & & \\ & \alpha_3 & 1 & & \\ & & & \ddots & \\ 0 & & \alpha_m & & 1 \end{pmatrix}$$

Clearly such a matrix is unimodular because it is lower triangular and all diagonal elements are 1. Moreover the first component of $C^{-1}x_B$ is h_i/d_i so that $g_1 = h_1$ and g_1 is integer. For components $i = 2,3,\ldots,m$, α_i must satisfy

$$\frac{g_i}{d_i} = \alpha_i \left(\frac{h_{i-1}}{d_{i-1}}\right) + \frac{h_i}{d_i} \qquad (3\text{-}6)$$

for some integers g_i. Any integer α_i will produce an integral g_i because d_{i-1} divides d_i and thus d_i is the least common denominator for the right-hand side of (3-6). Note that since C^{-1}, S and R^{-1} will all be integer matrices when generated in this way, $B = R^{-1}SC^{-1}$ is also integer.

The density of nonzero entries, the index of distance between the LP and IP solution, and integer feasibility are controlled in the generation of the nonbasic matrix N. The constraint matrix of ILP1 can be expressed as

$$Nx_N = b - Bx_B$$

Pick any column of N, say the first, and feasibility of ILP can be assured if we set

$$a = b - Bx'_B - N'x'_N \tag{3-7}$$

where b is as above, N = (a,N'), N' is an arbitrary, integer m x (n-m-1) matrix, and (x'_B,x'_N) is an arbitrary nonnegative integer (n-1)-vector.

Without loss of generality, x'_B may be the rounded version of x_B, the LP1 optimum solution. In (3-7), if x'_N is chosen to be an integral vector containing small values (say the first element is 1 and the rest are zero) it is conceivable that the "built-in" integer solution $(x'_B,1,x'_N)$ is very close to the LP optimum solution. It is therefore probable that this feasible solution is the ILP optimum solution. If x'_N has large values, it would be otherwise. The "distance index" parameter of the generator operates on this principle, using x'_N small for a low distance index and x'_N large for a high index.

Since the matrix N' is arbitrary, the density of nonzero entries in the complete matrix (B,a,N') can be largely set to the desired fraction by controlling the number on nonzero entries in N'. Note, however, that it is possible (though unlikely with judicious choice of R^{-1}) that (B,a) would have more nonzero entries than the total required for (B,a,N'), and in any event the generated constraint matrix is not homogeneous. Thus density is better controlled if n is at least twice m.

The cost vector c_B can be generated entirely randomly. To assure that x_B is the optimal LP solution it is only necessary to keep $(c_BB^{-1}N - c_N) \geq 0$, i.e.,

$$c_N \leq c_BB^{-1}N \tag{3-8}$$

Choice of c_N so that (3-8) holds as equality in the number of columns desired dual degenerate, and as a strict inequality in all other columns, completes the generation of the ILP problem. In cases where equality must hold may result in non-integral components of $C_B B^{-1} N$, c_B can be multiplied by the least common multiple of the denominators of such fractions so that c_N becomes integer.

3.2 The Generating Procedure

A generating procedure implementing the above rationale will now be stated. See Appendix A for a computer code of the procedure.

Step 1. For the specified number of constraints m generate m positive integers $d_1, d_2, \ldots, d_m$ such that d_i divides d_{i+1} for $i = 1, \ldots, m-1$, and $\pi_{i=1}^{m} d_i$ is the given determinant value D.

Step 2. Construct a diagonal matrix S such that the ith diagonal element is d_i, i.e.,

$$S = \begin{pmatrix} d_1 & & & 0 \\ & d_2 & & \\ & & \ddots & \\ 0 & & & d_m \end{pmatrix}$$

Step 3. Generate the LP optimal solution vector x_B. Set the ith entry of x_B equal to h_i/d_i, where the h_i are random nonnegative integers having a percent of zero elements equal to the specified degree of primal degeneracy.

Step 4. Generate the unimodular matrix C^{-1} and calculate $C^{-1}x_B$.

$C^{-1}x_B$ is required to satisfy:

$$\begin{pmatrix} g_1/d_1 \\ g_2/d_2 \\ \cdot \\ \cdot \\ \cdot \\ g_m/d_m \end{pmatrix} = C^{-1} \begin{pmatrix} h_1/d_1 \\ h_2/d_2 \\ \cdot \\ \cdot \\ \cdot \\ h_m/d_m \end{pmatrix}$$

for some integers $g_1, g_2, \ldots, g_m$. The requirement is achieved by taking

$$C^{-1} = \begin{pmatrix} 1 & & & & 0 \\ \alpha_2 & 1 & & & \\ & \alpha_3 & 1 & & \\ & & & \ddots & \\ & & & & \\ 0 & & \alpha_m & & 1 \end{pmatrix}$$

for random integers $\alpha_2, \alpha_3, \ldots, \alpha_m$.

Step 5. Compute $SC^{-1}x_B$ and denote the resulting (integer) vector by g.

Step 6. Randomly generate R as the product of random elementary matrices corresponding to the following elementary row and column operations:

i) Interchange rows (columns) p and q.

ii) Add an integer multiple of row (column) p to row (column q.

iii) Multiply row (column) p by -1.

Such a matrix R will be unimodular, so its inverse exists and is also unimodular. Calculate R^{-1}.

Step 7. Multiply g by R^{-1} on the left, to obtain

$$R^{-1}g = R^{-1}SC^{-1}x_B$$

Then set $B = R^{-1}SC^{-1}$ and $b = R^{-1}g$. The determinant of B has absolute value D, and both B and b are integer.

Step 8. Randomly generate a m x (n-m) preliminary submatrix N such that each column has at least one nonzero entry. The number n_1 of nonzero elements in N is

n_1 = (desired density of nonzero entries)(m)(n) - (number of nonzero entries in B).

Step 9. Set x_B' equal to the version of x_B obtained by rounding fractional components to the nearest integer. To partially control the distance between the optimal LP and IP solutions, generate an (n-m-1)-vector x_N' of positive integers with large magnitudes if distance is to be large and small magnitudes if distance is to be small.

Step 10. Revise one column of N, say the first, to achieve integer feasibility of ILP1. Generate this revised column (denoted a) as

$$a = b - Bx_B' - N\begin{pmatrix} 0 \\ x_N' \end{pmatrix}$$

Step 11. Generate c_B arbitrarily and multiply by a large enough constant (at most D) to make $c_BB^{-1}N$ integer.

Step 12. Generate $c_N = c_BB^{-1}N - \epsilon$, where ϵ is any nonnegative integer vector with the same proportion of 0 entries as the desired percent of dual degeneracy.

3.3 An Example

To help follow the generating procedure, an illustration is given in this section. All step numbers refer to the statement of the procedure in Section 3.2.

Suppose a problem is to be generated, with m = 3, n = 7, D = 16, primal and dual degeneracy = 0%, density = 50%, and distance index small.

Step 1: Let $d_1 = 2$, $d_2 = 2$, and $d_3 = 4$

$$\prod_{i=1}^{3} d_i = D = 16$$

Step 2 : Then

$$S = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 4 \end{pmatrix}$$

Step 3: Let $h_1 = 3$, $h_2 = 5$, and $h_3 = 5$. Therefore

$$x_B = \begin{pmatrix} 3/2 \\ 5/2 \\ 5/4 \end{pmatrix}$$

Step 4: For i = 2 the right-hand side of (3-6) is $\alpha_2(3/2) + (5/2)$. If $\alpha_2 = 1$, $g_2 = 8$. For i = 3, the right-hand side of (3-6) is $\alpha_3(5/2) + (5/4)$. If $\alpha_3 = 2$, $g_3 = 25$. So

$$C^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 2 & 1 \end{pmatrix}$$

Step 5: $g = \begin{pmatrix} 3 \\ 8 \\ 25 \end{pmatrix}$ because $g_1 = h_1 = 3$.

Step 6: Construct R from the three elementary row and column operations

$$R = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & -1 \end{pmatrix}$$

and

$$R^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 1 \\ 0 & -1 & 0 \end{pmatrix}$$

Step 7: The resulting B and b are

$$B = R^{-1}SC = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 1 \\ 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 4 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 2 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} 2 & 0 & 0 \\ -2 & 6 & 4 \\ -2 & -2 & 0 \end{pmatrix}$$

$$b = R^{-1}g = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 1 \\ 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} 3 \\ 8 \\ 25 \end{pmatrix} = \begin{pmatrix} 3 \\ 17 \\ -8 \end{pmatrix}$$

Step 8: Suppose the preliminary N matrix is randomly generated as

$$N = \begin{pmatrix} 0 & 0 & 2 & 0 \\ 1 & 3 & 5 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Note that the number of nonzero entries in (B,N) is 11 which is approximately 50% of m(n) = 21.

Step 9: The rounded x_B, is given by

$$x_B' = \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix}$$

Since distance index is to be small, let

$$x_N' = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

Step 10:

$$a = \begin{pmatrix} 3 \\ 17 \\ -8 \end{pmatrix} - \begin{pmatrix} 2 & 0 & 0 \\ -2 & 6 & 4 \\ -2 & -2 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix} - \begin{pmatrix} 0 & 0 & 2 & 0 \\ 1 & 3 & 5 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

$$= \begin{pmatrix} -1 \\ -5 \\ -3 \end{pmatrix}$$

Therefore the matrix N to be used is

$$N = \begin{pmatrix} -1 & 0 & 2 & 0 \\ -5 & 3 & 5 & 0 \\ -3 & 0 & 0 & 1 \end{pmatrix}$$

Step 11: From Step 7,

$$B^{-1} = \begin{pmatrix} 1/2 & 0 & 0 \\ -1/2 & 0 & -1/2 \\ 1 & 1/4 & 3/4 \end{pmatrix} ,$$

and

$$B^{-1}N = \begin{pmatrix} 1/2 & 0 & 0 \\ -1/2 & 0 & -1/2 \\ 1 & 1/4 & 3/4 \end{pmatrix} \begin{pmatrix} -1 & 0 & 2 & 0 \\ -5 & 3 & 5 & 0 \\ -3 & 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} -1/2 & 0 & 1 & 0 \\ 2 & 0 & -1 & -1/2 \\ -4-1/2 & 3/4 & 3-1/4 & 3/4 \end{pmatrix}$$

If $c_B = (4,4,8)$, $c_B B^{-1}N$ is an integral vector, i.e., $c_B B^{-1}N = (-30,6,26,4)$.

Step 12: $c_N = (-35,1,21,-1)$, where ε used is 5 in every component.

In summary, the ILP problem generated is

$$\text{Max} \quad 4x_1 + 4x_2 + 8x_3 - 35x_4 + x_5 + 21x_6 - x_7$$

$$\begin{aligned} \text{Subject to} \quad & 2x_1 - x_4 + 2x_6 = 3 \\ & -2x_1 + 6x_2 + 4x_3 - 5x_4 + 3x_5 + 5x_6 = 17 \\ & -2x_1 - 2x_2 - 3x_4 + x_7 = -8 \end{aligned}$$

$x_i \geq 0$ and integer for $i = 1,2,\ldots,7$.

CHAPTER IV

DEVELOPMENT OF AN EXPERIMENTAL APPROACH

As mentioned earlier, a great number of ILP studies attempt to both develop the theory of a proposed procedure and compare the efficiency of the procedure to that of other algorithms. Comparisons are usually made on the basis of differences in average solution time, average number of simplex iterations, etc. for some small number of test problems. In a strict sense conclusions drawn from such comparisons are valid only for the given test problems, and thus any broader conclusions must be stated in imprecise, "apparent tendency" terms.

When test problems are randomly generated--and thus solution times are realizations of a random variable--much stronger inferences can be made if due attention is paid to the conduct of the computational experiment. Appropriate use of the statistics of experimental design can make conclusions drawn from particular test problems valid for all randomly generated problems of given classes.

The techniques of statistical experimental design are well known and widely used in scientific research. However, a number of special considerations arise when they are applied to computational experiments on ILP procedures. The purpose of this chapter is to present and investigate analytically several important issues of this kind. In Chapter V these and related issues will be pursued from an empirical point of view. It is assumed throughout that test problems are randomly generated by a scheme like the one developed in Chapter III.

4.1 Selection of Response Variables

In an empirical study of ILP solution procedures, the first experimental design question to be answered is "What response variables should be used?" Several response variables have been suggested or reported, but none has evolved as a generally accepted standard. Most researchers have used solution times and/or numbers of various iterations in comparing the computational efficiency of algorithms. Others have suggested use of the amount of effort or money expended to obtain a solution (33). Still others have suggested that the number of elementary computer operations is a good measure of computational efficiency (50). While these latter two proposals might be well suited to comparisons of ILP algorithms in certain environments, they are not easily tabulated in the process of solving randomly generated problems. Thus, we will elaborate in this section only on the use of solution times, numbers of simplex iterations, and numbers of LP subproblems as response variables.

The total solution time of a LP-based ILP solution procedure can be expressed as:

$$\text{Total time} = \begin{Bmatrix}\text{Time for}\\ \text{initial LP}\end{Bmatrix} + \sum_{i=1}^{N} \begin{Bmatrix}\text{LP}\\ \text{restart}\\ \text{time } i\end{Bmatrix} + \begin{Bmatrix}\text{Time for post-opti-}\\ \text{mality analysis in}\\ \text{subproblem } i\end{Bmatrix} + \begin{Bmatrix}\text{Time}\\ \text{measurement}\\ \text{error}\end{Bmatrix} \qquad (4\text{-}1)$$

where N is the number of LP subproblems required for solution of the ILP. The "Time for initial LP" is the time needed to reach optimality in the linear programming relaxation of ILP. The second part of (4-1)

is the time needed to reach ILP optimality from LP optimality, i.e., the sum of the times required for the N, LP subproblems. Each such subproblem begins with the solution of a new linear program. For cutting plane algorithms that "LP restart time" is followed by a "Post-optimality time" required for generation of new cuts; for branch-and-bound algorithms, the post-optimality time is the time for developing improved bounds and searching for a new variable on which to branch. All times are subject to the accuracy of the computer clock and thus contain "Time measurement error".

Another measure of the efficiency of an LP-based ILP procedure is "total simplex iterations". In an analogous way to (4-1), total simplex iterations can be expressed:

$$\text{Total simplex iterations} = \begin{pmatrix}\text{Iterations for}\\ \text{initial LP}\end{pmatrix} + \sum_{i=1}^{N} \begin{pmatrix}\text{LP restart}\\ \text{iterations}\end{pmatrix}_i \qquad (4\text{-}2)$$

where N is defined as in (4-1). The "iterations for initial LP" is the number of simplex iterations required for optimality in LP. The second part of (4-2) is the total number of simplex iterations needed to achieve ILP optimality from LP optimality. It consists of the sum of the iterations required to restart the N, LP subproblems.

Both time for initial LP and LP restart time in (4-1) depend upon the LP code used in the experiment, the computing machine on which the algorithms are run, and the nuisance parameter characteristics of test problems. Similarly, time for post-optimality analysis in (4-1)

depends upon computing machines and nuisance parameters, and the various numbers of iterations in (4-2) depend upon LP codes and nuisance parameters.

In comparing ILP procedures the researcher's interest is primarily in differences in the efficiency of the "integer" aspects of the algorithm. Thus in a sense LP codes, computing machines, and nuisance problem parameters are all nuisances to the experimenter. The ideal experiment would provide control for all these concerns and make possible the isolation of an effect due only to the "integer efficiency" of the proposed algorithm.

Unfortunately, it is economically infeasible for most researchers to run experiments on a variety of computers and LP codes. Thus the analysis of this section will concentrate on experiments where only problem nuisance characteristics are controlled by the experimenter. Differences in LP codes and computing machines will be treated as experimental limitation which must be "normalized out" of experimental results.

It is well known that, as compared to ILP's, linear programs are relatively easy to solve. Thus it makes sense to concentrate the attention of an ILP researcher on the (presumably dominant) phase between linear and integer optimality in the solution of an ILP. Moreover, dropping the first term of either (4-1) or (4-2) leads to some reduction in variations due to different LP codes and computers because the wholly LP phase of the solution is omitted. Therefore, it is advantageous to use measures without the first term, i.e.,

$$\text{Total integer time} = \sum_{i=1}^{N} \begin{pmatrix} \text{LP} \\ \text{restart} \\ \text{time}_i \end{pmatrix} + \begin{pmatrix} \text{Time for post-} \\ \text{optimality analysis} \\ \text{of subproblem}_i \end{pmatrix}$$

$$+ \text{(Time measurement error)} \qquad (4\text{-}3)$$

and

$$\begin{pmatrix} \text{Total iterations} \\ \text{from LP to IP} \\ \text{optimality} \end{pmatrix} = \sum_{i=1}^{N} \begin{pmatrix} \text{LP restart} \\ \text{iterations}_i \end{pmatrix} \qquad (4\text{-}4)$$

Several strategies have been proposed to develop efficient ILP solution procedures. For our purposes, strategies are classified into two categories: (1) Strategies which attack the time for post-optimality analysis of subproblems, but hold the number of LP subproblems constant; and (2) strategies involving reductions in the number of subproblems. Since selection of response variables is somewhat different for these two categories of solution procedures, discussions on the subject are given separately.

4.1.1 Selection in Category 1

As mentioned above, category 1 is concerned with strategies with no interaction between the number of subproblems and the time for post-optimality analysis of subproblems. Such cases arise when procedural improvements are developed to reduce the time for post-optimality analysis of each subproblem without increasing the number of subproblems. An example of such an improvement would be the development of a new way to generate the same cutting plane.

In this category the total integer time of (4-3) and the total number of iterations from LP to IP optimality in (4-4) will provide

consistent algorithm ranking if the same LP codes and same computers are used. However, if results from different computers are compared, total integer time may lead to inconsistent ranking because LP restart time, time for post-optimality analysis of subproblems, and measurement error are all computer-dependent.

To offset the differences between different computers, some have suggested to use standard timing programs which adjust the times from one computer to another and put them all on the same basis. A typical Fortran standard timing program developed by Colville (11) simply inverts a 40 x 40 matrix ten times. Himmelblau (29) noted that there seems to be quite a discrepancy between differences in solution times adjusted for standardized times and actual differences in solution times for the same test problem on two different computers. Himmelblau's comment was based on experience with nonlinear programming procedures, but the observation seems applicable to ILP codes. Moreover, computer times are subject to the inherent variation of time measurement error. It has been reported that replicate results for the standard timing program obtained from identical computing hardware showed discrepancies of as much as 10% (12).

Based on the above discussion, total iterations from LP to ILP optimality might seem a preferable measure in the cross-computer case if the same LP code is used. However, (4-4) takes no account of post-optimality time which is sometimes substantial. One way around this limitation is to convert post-optimality times to "equivalent simplex iterations". Define the following equivalent measure:

$$\text{Total equivalent iterations from LP to IP optimality} = \sum_{i=1}^{N} \left\{ \text{LP restart iterations}_i + \frac{\text{Time for post-optimality analysis of subproblem}_i}{\text{Average time for 1 simplex iteration}} \right\} \quad (4\text{-}5)$$

where N is the number of subproblems. The "Average time for 1 simplex iteration" is defined as the ratio of the total of LP restart times to the total of LP restart iterations. Post-optimality time is thus expressed in terms of the equivalent number of simplex iterations. For cross-computer comparisons with the same LP code this measure appears to be the best of the three presented because it reduces both the timing error and computer-dependence of the time measure in (4-3) while adjusting for the absence of post-optimality times in (4-4).

An analogous problem to the difficulty in comparing results across computers arises when the effect of LP codes on the computational efficiency of category 1 strategies is considered. Differences in (4-3), (4-4), or (4-5) may be a consequence of the efficiency of the LP codes, rather than the effectiveness of the integer programming techniques employed.

When the same computer is used for all experimental observations, the obvious solution to this difficulty is to consider post-optimality time alone as the response variable. Thus algorithms would be compared on the basis of

$$\text{Total post-optimality time} = \sum_{i=1}^{N} \left\{ \text{Time for post-optimality analysis of subproblem}_i \right\} \quad (4\text{-}6)$$

However, the effect of LP codes is not so easily discounted

when different computers are used. The iteration normalizations of (4-3) and (4-4) can no longer be meaningfully employed. In this case it appears the only workable normalization is the use of standard times like Colville's (12).

4.1.2 Selection in Category 2

Category 2 is concerned with strategies which have interaction between the number of subproblems and the time for post-optimality analysis of subproblems. If strategies seek to reduce number of LP subproblems (i.e., N) they usually do so by increasing post-optimality analysis; likewise, if strategies seek to reduce post-optimality time, they usually do so by increasing N. Consider, for example, a cutting plane algorithm. It would be advantageous to reduce the number of LP subproblems by providing deeper cuts; however, it typically takes a longer time (i.e., post-optimality time) to obtain deep cuts.

Since ILP algorithm design often impacts on the number of LP subproblems, experiments of the computational efficiency of category 2 strategies are undoubtedly more common than those on category 1 strategies. However, the advantages and disadvantages of solution times and iteration counts in cross-computer comparisons hold equally well for categories 1 and 2. Thus most of the previous analysis is applicable to category 2.

The additional complicating factor in experiments on category 2 strategies is the role of the efficiency of the linear programming algorithm. Since both the number of LP subproblems and the time for post-optimality analysis may vary with algorithms, the simplifications of (4-6) cannot be employed. Elimination of LP restart time in (4-3)

would severely bias results in favor of a procedure which minimized post-optimality time. Moreover, there does not appear to be a simple alternative normalization for the efficiency of the LP code because the efficiency enters (4-3), (4-4), and (4-5) in a multiplicative way. The amount of impact the code has depends on the magnitude of N.

For this reason it appears that experimenters investigating category 2 strategies are forced to record both the total time or iterations of (4-3) and (4-5), and the number of LP subproblems, N. These two response variables may occasionally produce conflicting rankings, but no resolution of this conflict is apparent. Comparisons would be based primarily on (4-3) or (4-5), but tabulation of N provides some compensation for the effect of the LP code.

The problem becomes even more complex when different computers are used in experiments. The conversion to simplex iterations embodied in (4-5) is only meaningful if the time per iteration is relatively constant. In this very complex case, it appears the only realistic approach is the use of standard time normalizations like Colville's (11) along with the number of subproblems (N). However, any comparisons would be suspect.

4.1.3 Summary and Remarks

The selection of response variables discussed above is summarized in Table 1. Because it is apparently as good as any other measures for experiments on one computer, we will assume in the rest of the chapter that total integer time, i.e., (4-3), is selected as a response variable. However, we will return with empirical data on this issue in Chapter V.

Table 1. Selection of Response Variables

	Category 1 (Changes in post-optimality times only)		Category 2 (Changes in N)	
	Same LP	Different LP's	Same LP	Different LP's
Same computer	Total integer time (4-3) or equivalent iterations (4-5)	Total post-optimality time (4-6)	Total integer time (4-3) and LP sub-problems (N) or equivalent iterations (4-5) and LP subproblems (N)	Total integer time (4-3) and LP sub-problems (N)
Different computers	Equivalent iterations (4-5)	Post-optimality time (4-6) adjusted by standard timing factors	Equivalent iterations (4-5) and LP subproblems (N)	Total integer time (4-3) adjusted by standard timing factors and LP sub-problems (N)

4.2 Treatment of Censored Data

It is often reported in the literature that particular ILP test problems could not be solved within a specified time limit (see for example (54)). Similar experimental limitations also arise in life testing of machinery and electrical devices, and in biological experiments. For instance, the lives of some experimental motors and light bulbs may last longer than the experiment period. Similarly, the effect of a medical treatment on mice may not materialize within the experimental period. In each such case the experimental response value is not obtained, but some information is produced. The researcher knows the missing value falls outside some known lower or upper bound.

In the statistical literature such data are called censored data. Data are said to be singly censored if the values of observations in only one of the distribution tails cannot be obtained, and doubly censored if the values of the observations in both tails are unknown. Thus all the above examples, including ILP solution times, are singly censored to the right.

Censored data are said to be of Type I if censoring occurs on all observations falling outside specified values of the dependent variable; whereas, censored data are said to be of Type II if the number of censored observations is specified and their censored values are random. When censoring of ILP solution times occurs because a maximum time limit is reached, the data are Type I censored. An example of Type II censoring is an experiment where k light fixtures are kept running, new bulbs being inserted whenever old ones fail. Exactly k failure time values will be censored at the end of the experiment.

A number of alternatives have been proposed for treating censored data in the experimental design context. One alternative is to consider censored data as missing and use the usual procedure for missing experimental data to do statistical analyses. Such procedures include replacing the missing value with a hypothetical one which minimizes the sum of squares for error, and solving a reduced set of least squares normal equations for the experiment which omit the missing values.

The missing data approach does not seem desirable in the ILP experiments. In our experience with the cutting plane algorithm used in this study, test problems are solved in two distinctive ways, most

being solved in very short time, but some remaining unsolved due to the numerical difficulty. Therefore the estimates for censored data would be significantly biased toward low values if the above procedure were followed because all analysis would be based on the short, uncensored times. Moreover, this approach would discard all the information we have about the most troublesome test problems, those with long solution times. Finally, in cells where the nuisance parameters are set at high values, no data may be observed, no matter how many replicates were run. Missing data schemes are particularly ineffective in dealing with cells where all data are missing.

A second alternative for dealing with censored data is to treat the censor points (usually the time limits) as observed data and do the statistical analyses as usual. This method does take some advantage of our information of hard problems, but clearly biases results toward low values.

A third alternative for treating censored data is to use as the estimate for each unknown point a computed value that appears most probable on the basis of the available data. A number of statistical studies have developed estimates of this kind for various types of censored data, especially for distributions arising in life-testing situations. The underlying distributions used include exponential, normal, and Weibull distributions. Nelson and Hahn (42, 43) give a review of such studies and make a comparison of statistical methods proposed. For the Type I censoring case occurring in ILP experiments their analysis favors the approach of using maximum likelihood estimates of the censored values.

Sampford and Taylor (47) developed the only known application of this maximum likelihood method in an experimental design setting, randomized block experiments with Type I censored observations. Later Taylor (50) conducted a large empirical test of Sampford and Taylor's method and concluded that the technique works well. Since the maximum likelihood method appears to be best for Type I censored and the experimental designs, the method of Sampford and Taylor is the one adopted in this dissertation. The remainder of this section concentrates on adapting their approach to the case of ILP computational experiments.

4.2.1 Sampford and Taylor's Method for Randomized Block Designs

In a randomized block experiment with I blocks and J treatments Sampford and Taylor assume that the observation for treatment j in block i is given by

$$y_{ij} = \mu_{ij} + \varepsilon_{ij} \quad (i = 1,\ldots,I \; ; \; j = 1,\ldots,J)$$

where $\mu_{ij} = \mu + \alpha_i + \beta_j$ (with $\Sigma\alpha_i = \Sigma\beta_j = 0$) and where the ε_{ij}'s are normally and independently distributed with zero mean and variance $\sigma^2 = 1/\gamma^2$. These are the standard assumptions for such experiments, but the fact that normality and equality of variance will be explicitly required in the development presumably means that adjustments in the data would be required if such assumptions were significantly violated. We will return to this issue in Chapter V.

From any set of values for the y_{ij},μ_{ij} in the above is estimated by

$$\hat{\mu}_{ij} = y_{i.} + y_{.j} - y_{..} \qquad (4\text{-}7)$$

where a dot in a subscript indicates averaging over that subscript.

Now suppose that C of the IJ observations are Type I censored at value U_{ij} respectively, i.e.,

$$y'_{ij} > U_{ij}$$

for the C observations, where ' denotes censored data. The probability of the inequality $(y_{ij} > U_{ij})$ being satisfied when y_{ij} is normally distributed is then

$$Q_{ij} \triangleq Q(\eta_{ij}) \triangleq \frac{1}{\sqrt{2\pi}} \int_{\eta_{ij}}^{\infty} e^{-\frac{1}{2}u^2}\, du$$

where $\eta_{ij} = (U_{ij} - \mu_{ij})/\sigma$.

Thus the likelihood function is

$$L = \pi \frac{\gamma}{\sqrt{2\pi}}\ e^{-\frac{1}{2}\gamma^2(x-\mu_{ij})^2}\ \pi' Q_{ij}$$

where π denotes the product over all pairs (i,j) for which the observation is not censored and π' denotes the product over pairs for which the observation is censored.

Following the usual maximum likelihood procedure with minor modifications due to censored data, Sampford and Taylor showed that maximum likelihood estimates for y'_{ij} are given by

$$\hat{y}'_{ij} = \hat{\mu}_{ij} + \hat{\sigma}\nu(\hat{\eta}_{ij}) \tag{4-8}$$

where

$$\nu(\hat{\eta}_{ij}) = f(\hat{\eta}_{ij})/\{1 - F(\hat{\eta}_{ij})\},$$

$f(\hat{\eta}_{ij})$ and $F(\hat{\eta}_{ij})$ are respectively the frequency and distribution function of a standardized normal variate,

$$\hat{\eta}_{ij} = (U_{ij} - \hat{\mu}_{ij})/\hat{\sigma} \ , \tag{4-9}$$

and

$$\hat{\sigma}^2 = \Sigma(y_{ij} - \hat{\mu}_{ij})^2/\{IJ - C + \Sigma\lambda(\hat{\eta}_{ij})\}. \tag{4-10}$$

Both summations in (4-10) are over all i and j, both censored and observed, and

$$\lambda(\hat{\eta}_{ij}) = \nu(\hat{\eta}_{ij})\{\nu(\hat{\eta}_{ij}) - \hat{\eta}_{ij}\}$$

To obtain estimates of the y'_{ij} calculations are done iteratively as follows:

(a) Initial values of y'_{ij} are assumed, typically the U_{ij}.

(b) The estimates $\hat{\mu}_{ij}$ of the cell means μ_{ij} are calculated from equation (4-7), using the C assumed values and (IJ-c) exactly observed values of y_{ij}.

(c) $\hat{\eta}_{ij}$ and $\hat{\sigma}^2$ are then calculated from equation (4-9) and (4-10).

(d) A set of revised values of y'_{ij} is obtained from equation (4-8), holding the μ_{ij} and $\hat{\sigma}$ constant.

(e) This set of revised values y'_{ij} is used to revise $\hat{\sigma}^2$.

Step (b) through (e) are repeated until both y'_{ij} and $\hat{\sigma}^2$ converge.

When no observations are censored, maximum likelihood estimation gives unbiased estimates of α_i and β_j. For $\hat{\sigma}^2$ it gives a biased

estimate, obtained by dividing the residual sum of squares by IJ, whereas it should be divided by the degrees of freedom (I-1) (J-1) to obtain an unbiased estimate. For experiments with censored values, it is evident that the maximum likelihood estimate of σ^2 will be similarly biased. Analogy with experiments having no censored observations suggests that the replacement of the divisor in (4-10) by

$$(I-1)\ (J-1) - C + \Sigma\lambda(\hat{\eta}_{ij})$$

would approximately correct for the bias. Experiments in (52) showed that the correction indeed reduced the bias of σ^2.

4.2.2 The Procedure for Full Factorial Designs

The randomized block design, for which Sampford and Taylor's procedure was devised, can be viewed as a single replicate of a factorial design, i.e., one set of say K sets of data, each containing one observation per (i,j) cell. The full experimental model for K replicates is given by

$$y_{ijk} = \mu_{ij} + \varepsilon_{ijk}$$

where $\mu_{ij} = \mu + \alpha_i + \beta_j + (\alpha\beta)_{ij}$ and there are i = 1,2,...,I levels of a factor corresponding to the α_i; j = 1,2,...,J levels of a factor corresponding to the β_j; and k = 1,2,...,K replicates. Here $(\alpha\beta)_{ij}$ is a term reflecting the interaction between the ith level of the first factor and the jth level of the second.

Preliminary experience--confirmed by the empirical evidence in Chapter V--showed that the residual variance in computational experiments on ILP's is quite large. Thus it seems important to be able to

deal with the case where several replicates of an experiment are run to obtain more powerful tests of the significance of the factors.

Sampford and Taylor noted that although their procedure was restricted to randomized block experiments, the computational routine could be extended to the analysis of any design for which the method of maximum likelihood leads, for experiments with complete data, to estimates of location parameters which are linear functions of the observations. Since the randomized block design can be considered as one replicate of a factorial design, where the factor combinations of the factorial design are the "treatment" and "block" of the randomized block design. It was concluded that the Sampford and Taylor procedure can be applied to the full factorial case.

Recall that at each iteration the Sampford and Taylor procedure for the randomized block design obtains new estimates of the cell means μ_{ij} by the averaging formula (4-7). These means then form the basis for the revision of estimates for the censored values in (4-8). The new estimates are in turn used to estimate the residual variance σ^2 at (4-10).

For the full factorial case the estimate of the (i,j) cell mean in terms of fixed y_{ijk} is given by

$$\mu_{ij} = y_{ij.}$$

where as before a dot subscript indicates averaging over all values with that subscript. An adaption of the Sampford and Taylor procedure to the factorial case would thus use this formula to estimate the cell means at each iteration. Notice that this estimate depends only on

the observations in the (i,j) cell. All other y_{ijk} are eliminated from the average by the choice of the estimate for the interaction $(\alpha\beta)_{ij}$.

Consider now a cell (i,j) where all y_{ijk} are censored and suppose the estimates of the censored values are

$$y'_{ijk} = v_{ij}$$

at some iteration of the Sampford and Taylor procedure. Then the estimate of the cell mean for that iteration will be

$$\mu_{ij} = v_{ij} \tag{4-11}$$

This implies that cell (i,j) will make no contribution to $\hat{\sigma}^2$ in (4-10). It follows that $\hat{\sigma}^2$ will be uneffected from iteration to iteration by changes in the y'_{ijk} and estimates of the y'_{ijk} will not converge. If the y'_{ijk} were not all equal at the beginning of the procedure, $\hat{\sigma}^2$ would be reduced until they became equal, but would be unchanged beyond that time.

The above argument shows that a direct generalization of the Sampford and Taylor procedure to the full factorial case breaks down when all values in a cell are censored. Since it is highly likely that all values would be censored at high levels of the nuisance parameters in computational experiments, such a generalization appears unworkable for computational experiments.

One way around this difficulty is to apply the Sampford and Taylor procedure of Section 4.2.1 to one replicate at a time, considering each replicate of the factorial design as a randomized block

design. Two observations regarding this proposed procedure are in order. The procedure makes multiple censored values in the same cell have different estimates in different replicates, which is intuitively appealing. The second observation is that censored data are estimated one replicate at a time, which may cause a replication effect in the experiment. Replications essentially become new blocks and the analysis of the experiment must be adjusted accordingly.

4.3 Application of the Analysis of Variance

For many years one of the most commonly used tools in the analysis of experimental data has been the analysis of variance (ANOVA). The ANOVA method has been employed to analyze data arising in agriculture experiments, industrial production experiments, and many other experimental situations. The principle of the technique is that if a set of observations can be classified according to one or more criteria, then the total variation between the members of the set can be broken up into components which can be atrributed to the different criteria of the classification. By testing the significance of these components it is then possible to determine which of the criteria are associated with a significant proportion of the overall variation. To carry out the analysis it is necessary to assume for the data a model which involves a number of parameters and properties, and a part of the technique consists of estimating these parameters.

As briefly outlined in Chapter I, this dissertation is addressed to the design and analysis of computational experiments where the research goal is to compare effectiveness of ILP solution procedures at different values of nuisance variables (recall that nuisance

variables are the parameters of problem types which tend to make some problems hard and others easy. Since computational experiments of this type fit naturally into the ANOVA setting, the basic tool of all data analysis in this dissertation will be the analysis of variance. The remainder of this section discusses the adaption of ANOVA to ILP computational experiments.

4.3.1 Verification of ANOVA Assumptions

One factor largely affecting the power of the ANOVA technique is the accuracy of the assumptions in the model underlying the procedure. The statistical model of the ANOVA technique is linear in all parameters. Since some have observed that the solution times of some ILP solution procesures grow exponentially with the number of discrete variables (20), it appears some examination of ILP experimental data is appropriate to see whether the assumptions of ANOVA are sufficiently satisfied. In this chapter we will briefly review these assumptions, methods for testing their appropriateness, and methods for transforming data to obtain a suitable fit to the assumptions. In Chapter V we will return to this issue by empirical application of such tests and transforms.

To facilitate the discussion of the ANOVA assumptions, let us assume the simple one-way classification model that for each observation

$$x_{ij} = \mu + \alpha_i + \varepsilon_{ij} \quad (i = 1,\ldots,I \; ; \; j = 1,\ldots,J_i) \quad (4\text{-}12)$$

i.e., each observation is the sum of an overall mean μ, an effect α_i due to the class in which the observation occurs, and a residual ε_{ij}

which represents the variation of the particular value x_{ij} from the average value of the ith class. The ε_{ij} are assumed to be independently and identically normally distributed with mean 0 and variance σ^2. In the following, these assumptions are briefly discussed, pertinent tests of assumptions reviewed, and the remedial steps to be taken presented. It should be noted that these assumptions are not equally important. See Eisenhart (17), Cochran (11), Bartlett (2), etc. for summaries of the consequences of failure to satisfy particular assumptions.

4.3.1.1 The Assumption of Independence of Residuals (ε_{ij}). The ANOVA procedure assumes the residual terms (ε_{ij}) are mutually independent random variables. Failure in this assumption invalidates the technique completely. However, in the ILP setting where test problems are randomly generated, this difficulty should not arise.

4.3.1.2 The Assumption of Normality of the Residuals (ε_{ij}). The ANOVA method also assumes the residuals are normally distributed. If the residuals (ε_{ij}) are not normally distributed the true significance of hypothesis tests will not be exactly the one tabulated. However, it is well known (see for example Davies (14)) that the F-tests implicit in the ANOVA procedure are robust to the normality assumption. In ILP experiments whose object is to compare solution means, F-tests will provide an adequate approximation even when fairly large departures from normality occur.

The tests of distribution such as the Chi-square test and the Kolmogorov-Smirnov test are usually used to check the normality assumption. The remedial step for correction of serious departures from the assumption is the use of data transformation, which is to be discussed

later in this section.

4.3.1.3 The Assumption of the Equality of Residual Variance. The third ANOVA assumption about the residuals is that they have equal variance at all levels of the experimental classification. If the variance of the residual differs from one observation to another, the usual method of analysis leads to a loss of efficiency in the estimation of effects and a distortion of the significance level of ANOVA comparisons. Thus in contrast to the normality assumption the equal residual variances assumption is crucial to the usefulness of ANOVA.

One case where the assumption would not be satisfied is when the variance of observations in various classes is a function of the class mean. In ILP experiments where classification was partially based on number of discrete variables, we might very well expect solution time variances to grow with problem size. Thus we will here treat tests and remedies for the equal variance assumption in some detail.

The Bartlett (3) test is an old, well-established test that has been used to test the equality of variances, but it is known to be sensitive to non-normality (5). The test is especially sensitive to non-normality if the tails of the distribution are long. In this case the Bartlett test tends to reject the hypothesis of equal variance too often.

Several tests for the equality of variances, which are less sensitive to non-normality, have been proposed and a comparison of these tests may be found in Brown and Forsythe (9). The one of these robust tests for the equality of variances selected for use in this dissertation is Levene's procedure. Levene (39) proposed the following statistic, using the model in (4-12).

Let $z_{ij} = |x_{ij} - x_{j.}|$. Then the test statistic is

$$W = \frac{\sum_i J_i (z_{i.} - z_{..})^2 / (I - 1)}{\sum_i \sum_j (z_{ij} - z_{i.})^2 / \sum_i (J_i - 1)}$$

where a dot subscript again indicates averaging over all values of that subscript. The critical values of W are obtained from the Snedcor F-table with I-1 and $\sum_i (J_i - 1)$ degrees of freedom.

The reasoning behind Levene's test is the following. The z_{ij}, $j = 1,2,\ldots,J_i$ are all estimates of some multiple of the cell residual variance, σ_i, for $i = 1,2,\ldots,I$. As mentioned earlier, the ANOVA procedure is known to be very robust for testing differences between means of groups. We then test, robustly, whether differences exist between the means of the I sets of estimates. If not we conclude that there is no evidence against the hypothesis that $\sigma_1^2 = \sigma_2^2 = \ldots = \sigma_I^2$.

If a test of the equal variance assumption has suggested inequality of variance, a number of alternatives are available. At worst it may be necessary to proceed with the analysis in the usual way, interpreting the apparent conclusions with more or less reserve. Another procedure which may be desirable in cases where a sufficient number of replicates are available in each cell is to weight each observation in proportion to the inverse of its error variance. This procedure postulates, however, a knowledge of the relative variances of any two observations and this knowledge is seldom available in practice. Still, another procedure often used to obtain a constant error variance is the use of transformation. As mentioned earlier,

the introduction of a transformation also may be necessary to make the error distribution normal. Since the transformation alternative seems most appropriate for computational experiments a discussion of data transformation is given below.

Due to its importance in the data analysis, a number of methods have been proposed to choose a suitable transformation (2, 15, 16). If the variance is a known function of the mean, numerous special transformations for use have been examined in Bartlett (2). The search for a transformation, however, is first treated generally by Box and Cox (7).

The principle of the Box and Cox technique is to find a transformation which maximizes the likelihood that the transformed data arose from a process with normally, independently, and identically distributed residuals. Following their development, suppose that we observe a dependent variable y and that the appropriate linear model for the problem is specified by

$$E\{y^{(\lambda)}\} = A\theta$$

where $y^{(\lambda)}$ is the column vector of transformed observation, λ is a parameter which characterizes the transformation, A is a known matrix, and θ is a vector of unknown parameters associated with the transformed observations. Assume that for some unknown λ, the transformed observations $y_i^{(\lambda)}$ $(i = 1,\ldots,n)$ satisfy the normal theory assumptions, i.e., are independently normally distributed with constant variance σ^2, and with expectations A . Then the likelihood function for the observations $y^{(\lambda)}$, in logarithmic form, is

$$\log L(\lambda) = -\frac{1}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}(y^{(\lambda)} - A\theta)(y^{(\lambda)} - A\theta)$$

$$+ \log J(\lambda;y)$$

where $J(\lambda;y)$ is the Jacobian of the inverse transformation taking $y^{(\lambda)}$ to the actually observed y.

For fixed λ, the maximized log likelihood is, except for a constant,

$$L_{max}(\lambda) = -\frac{1}{2}\log\hat{\sigma}^2(\lambda) + \log J(\lambda;y)$$

where $n\hat{\sigma}^2(\lambda)$ is the residual sum of squares in the ANOVA of $y^{(\lambda)}$. The function $L_{max}(\lambda)$ can be shown to be a unimodal function of λ. Thus the best value of λ may be determined by plotting $L_{max}(\lambda)$ against λ for a trial series of values. From this plot the maximizing value $\hat{\lambda}$ may be read off and an approximate $100(1-\alpha)$ percent confidence region on $\hat{\lambda}$ is obtained by the standard approach as

$$L_{max}(\hat{\lambda}) - L_{max}(\lambda) < \frac{1}{2}\chi^2_{\nu_\lambda}(\lambda)$$

where ν_λ is the number of independent components in λ.

The $y^{(\lambda)}$ in the Box-Cox procedure is a general monotonic function of y. A family of transformations of particular importance is

$$y^{(\lambda)} = \begin{cases} y^\lambda & \lambda \neq 0) \\ \log y & (\lambda = 0) \end{cases}$$

It is clear that this transform includes as special cases $\sqrt{y}$, y^{-1}, $\log y$,

etc. which are sometimes recommended for y with particular distributions.

4.3.2 ANOVA with Censored Observations Estimated

The analysis of variance with censored observations estimated is an interesting, but unexplored area. As discussed in Section 4.2.2, the type of ILP experiments of interest is a factorial design with Type I censoring. Little work has been found on the treatment of such cases in the statistical literature.

In the case of randomized block experiment with Type I censoring, Sampford and Taylor (49) noted that exact tests of significance of treatment differences could be made only if the sampling variances and covariances of the treatment effects, β_j, and the sampling distribution of the modified variance estimate $\hat{\sigma}^2$, were known. These variances are mathematically intractable. Thus Sampford and Taylor reasoned by analogy to the uncensored case to develop an approximate t-test.

If C of the I observations for a treatment are censored, the variance of the corresponding $\hat{\beta}_j$ will be greater than $\hat{\sigma}^2/I$, but less than $\hat{\sigma}^2/I-C$. It seems reasonable to take it to be

$$\text{Var } (\hat{\beta}_j) = \frac{\hat{\sigma}^2}{I - C + \sum_i \hat{\lambda}_{ij}}$$

where $\hat{\lambda}_{ij}$ are as developed in Section 4.2.1 and the summation is over all $\hat{\lambda}_{ij}$ for treatment j. Here $\hat{\lambda}_{ij}$ is being used as a measure of the contribution of the censored observation to the treatment mean.

If the percentage of censored observations is not great, correlations of the $\hat{\beta}_j$ for different treatments would be negligible. Then

$$\text{Var}\ (\hat{\beta}_j - \hat{\beta}_{j'}) = \text{Var}\ (\hat{\beta}_j) + \text{Var}\ (\hat{\beta}_{j'}) \qquad (4\text{-}13)$$

Since the t-statistic is $t = (\hat{\beta}_j - \hat{\beta}_{j'})/\text{Var}\ (\hat{\beta}_j - \hat{\beta}_{j'})$, we can then carry out an approximation to a t-test, using as "degree of freedom" the divisor used in obtaining the modified estimate of $\hat{\sigma}^2$.

In (49) a study of the variance-covariance matrix of $\hat{\beta}_j$ estimates was made to determine the possible effects of correlation between the means on the approximate tests. Empirical results indicate that the use of formula (4-13) rarely increases the magnitude of t-statistic by more than 10 percent.

It appears reasonable to extend the analogy to the other ANOVA procedures and the F-test. However, it must be stressed that a value near an operative significance level of the F-distribution should be accepted as significant at that level only with reservations.

4.4 Selection of Experimental Design

As mentioned in Chapter I, the objectives of this dissertation research are two-fold:

(1) Identification of parameters of ILP problems which significantly correlate with the computational complexity of ILP problems; and

(2) Development of controlled experiments for testing of proposed ILP solution procedures.

This section is aimed at determining experimental designs which most suitably achieve the above objectives.

Classical discussions of the principles of experimentation emphasized the importance of varying the factors in an experiment

in order to observe the effect upon the dependent variables being studied. Factors tended to be varied one at a time, rather than all the factors in combination. In two respects, however, this one-at-a-time approach is now generally seen to be inadequate. Firstly, this approach can have no hope of evaluating the interactions between factors. Not only does this deprive us of essential knowledge of the relationships between factors; it may actually be positively misleading. Secondly, this approach may cause the danger of attributing to one or more of the experimental factors, effects on the dependent variable which are in reality not due to these factors, but are due to variations in some factors not included in the experiment. The design eliminating such difficulty is the factorial design.

In the preceding section the effects of departures from the assumptions on the ANOVA were discussed. A good selection of experimental design, may facilitate statistical analyses and provide improved statistical accuracy. As Box (6) pointed out, designs with equal frequencies in all cells of the classification have two advantages: (i) computations are made much easier, and (ii) the effects of inequality of error variances are not serious. Thus the discussion of this section will be limited to factorial designs with equal frequencies in all cells.

Recall from Chapter I that the type of computational experiments which motivate all the analysis of this dissertation are those where several ILP algorithms can be arrayed against classes of test problems characterized by nuisance parameters. Examples of nuisance parameters are the problem parameters controlled by the problem

generation procedure developed in Chapter III (number of variables, number of constraints, determinant of the optimal LP basis, etc.)

Suppose for simplicity that we are concerned with only one such nuisance parameter at two levels and two possible algorithms. Such an experiment is schematically represented in Table 2.

Table 2. Basic Experimental Arrangement for a 2 by 2 Example

	Nuisance Parameter Level 1			Nuisance Parameter Level 2
Algorithm 1	Problem 1	. . .	Problem k	. . .
Algorithm 2	. . .			. . .

The equal frequency, factorial design approach to such an experiment would randomly generate and solve a series of say k problems in each cell represented in Table 2. For each problem the solution time required by the ILP algorithm would be observed and the analysis of variance used to determine whether algorithms were significantly different.

Note, however, that experiments arranged in this way use problems which cannot be compared across different levels of the nuisance parameter. Within a given level of the nuisance parameter there will

be a variation among randomly generated problems, but this variation can only be understood with respect to the particular level of the nuisance parameter. Moreover, if different problems are used in each cell, the variation due to problems will also be incomparable across algorithms.

Such an experimental phenomenon is known in the literature of experimental design as nesting. Thus we see that our computational experiments in fact involve three main effects, in addition to residual error. Variance among solution times is viewed as being caused by (i) variation among algorithms, (ii) variation among levels of nuisance parameters, and (iii) variation between problems generated within levels of algorithms and nuisance parameters. The first two effects are fixed effects and the problem effect is random.

Even within the format of Table 2, two factorial design approaches are reasonable. One design (which will be referred to as Case I) uses the same test problems on both algorithms. The experimental layout for this design is illustrated in Table 3. In the terminology of experimental design this Case I approach is called blocking on problems. Since a problem is fully defined by the seeds used to initialize the pseudo-random number routines in the problem generator, an equivalent terminology is blocking on random seeds. Blocked experiments are widely used in agricultural and industrial experimental settings in order to minimize the interference in ANOVA results caused by differences in raw materials, test procedures, and similar restrictions on randomization.

Note that under the blocked approach of the Case I design for ILP experiments the problem effect <u>can</u> be compared across algorithms, but not across nuisance parameters. Thus the problem effect is nested

Table 3. Experimental Layout for Case I: Blocking on Ramdom Seeds

	N_1		N_2	
	P_1	P_2	P_3	P_4
A_1	Prob a	Prob b	Prob c	Prob d
A_2	Prob a	Prob b	Prob c	Prob d

Table 4. Expected Mean Squares for Case I: Blocking on Ramdon Seeds

	I F	J F	K R	1 R	EMS	DF
A	0	J	K	1	$\sigma_\epsilon^2 + \sigma_{AP(N)}^2 + JK\sigma_A^2$	I-1
N	I	0	K	1	$\sigma_\epsilon^2 + I\sigma_{P(N)}^2 + IK\sigma_N^2$	J-1
AN	0	0	K	1	$\sigma_\epsilon^2 + \sigma_{AP(N)}^2 + K\sigma_{AN}^2$	(I-1)(J-1)
P(N)	I	1	1	1	$\sigma_\epsilon^2 + I\sigma_{P(N)}^2$	J(K-1)
AP(N)	0	1	1	1	$\sigma_\epsilon^2 + \sigma_{AP(N)}^2$	J(I-1)(K-1)
ϵ	1	1	1	1	σ_ϵ^2	0

Note: I, J, and K are levels of factors A, B, and P respectively; σ_A^2, σ_N^2, σ_{AN}^2, $\sigma_{P(N)}^2$, $\sigma_{AP(N)}^2$, and σ_ϵ^2 are the variance of the effect as indicated in the subscript.

only in nuisance parameters and the statistical model for Case I is

$$y_{ijk} = \mu + A_i + N_j + AN_{ij} + P_{k(j)} + AP_{ik(j)} + \epsilon_{k(ij)}$$

where y_{ijk} = the response on replicate k at nuisance level j when solved by algorithm i

μ = the overall mean response

N_j = the effect due to the jth nuisance parameter level

AN_{ij} = the effect due to algorithm interaction in cell (i,j)

$P_{k(j)}$ = the effect due to problem (i.e., replicate) k within the jth level of nuisance parameters

$AP_{ik(j)}$ = the effect due to algorithm vs. problem interaction for algorithm i and problem k

$\epsilon_{k(ij)}$ = the residual error in cell (i,j) on replicate k.

One immediate difficulty apparent in the model is that the various $AP_{ik(j)}$ and $\epsilon_{k(ij)}$ are observed on exactly the same y_{ijk}. Thus the variation due to algorithm-problem interaction cannot be separated from residual error unless an independent estimate of error variance is available. However, this confounding does not deprive us of the ability to construct tests for the significance of either the main algorithm effect (A), or the main nuisance parameter effect (N), or the algorithm-nuisance parameter interaction (AN). Table 4 presents the expected mean square table for the blocked case. From the table it follows that the proper test statistics are

$$F_A = \frac{SS_A/(I-1)}{SS_{AP(N)}/(I-1)(J)(K-1)}$$

$$F_N = \frac{SS_N/(J-1)}{SS_{P(N)}/J(K-1)}$$

$$F_{AN} = \frac{SS_{AN}/(I-1)(J-1)}{SS_{AP(N)}/(I-1)J(K-1)}$$

where SS denotes the sum of squares for the effect shown as a subscript.

It is worth noting that if it were desired to separate the residual error variance from the AP(N) interaction, it would probably be quite easy to obtain an independent error of residual variance. In ILP computational experiments, the only likely source of residual error is in the measurement of the response variable. For example, (see Section 4.1) variations in the internal clock of a computer may produce some random variation in solution times. The size of such a variation could be easily estimated by solving a few problems repeatedly and recording the differences in solution times reported by the computer.

The second possible design for an algorithm versus nuisance parameter experiment is the completely randomized design where different problems are used in each cell and each replicate. The layout for such a randomized design is illustrated in Table 5. In this design (which will be referred to as Case II), the problem effect is nested in both algorithms and nuisance parameters because problems cannot be compared across algorithms. Thus there can be no algorithm-problem interaction and the model for Case I collapses to

Table 5. Experimental Layout for Case II: Randomized Design

	N_1		N_2	
A_1	Prob a	Prob b	Prob e	Prob f
A_2	Prob c	Prob d	Prob g	Prob h

Table 6. Expected Mean Squares for Case II: Randomized Design

	I F	J F	K F	EMS	DF
A	0	J	K	$\sigma^2_\varepsilon + \sigma^2_{P(AN)} + JK\sigma^2_A$	I-1
N	I	0	K	$\sigma^2_\varepsilon + \sigma^2_{P(AN)} + IK\sigma^2_N$	J-1
AN	0	0	K	$\sigma^2_\varepsilon + \sigma^2_{P(AN)} + K\sigma^2_{AN}$	(I-1)(J-1)
P(AN)	1	1	1	$\sigma^2_\varepsilon + \sigma^2_{P(CAN)}$	IJ(K-1)
ε	1	1	1	σ^2_ε	0

Note: Notation is as in Table 4.

$$y_{ijk} = \mu + A_i + N_j + AN_{ij} + P_{k(ij)} + \epsilon_{k(ij)} .$$

From the model it is apparent that in Case II the problem effect is confounded with residual error. As noted above, however, problem variance could fairly easily be estimated by obtaining an independent estimate of residual variance.

Table 6 presents the expected mean squares for the Case II design. It is clear that the proper test statistics for the A, N, and AN effects are as follows:

$$F_A = \frac{SS_A/(I-1)}{SS_{P(AN)}/IJ(K-1)}$$

$$F_N = \frac{SS_N/(J-1)}{SS_{P(AN)}/IJ(K-1)}$$

$$F_{AN} = \frac{SS_{AN}/(I-1)(J-1)}{SS_{P(AN)}/(IJ(K-1)}$$

The efficiency of the blocked design (Case I) relative to that of the fully randomized design (Case II) can be measured by the ratio of the corresponding F-statistics. Consider first the algorithm (A) effect. The appropriate relative efficiency ratio (denoted $R.E._A$)) is given by

$$(R.E.)_A = \frac{\sigma^2_\epsilon + \sigma^2_{P(AN)}}{\sigma^2_\epsilon + \sigma^2_{AP(N)}}$$

If $R.E._A$ is greater than 1, i.e., $\sigma^2_{P(AN)} > \sigma^2_{AP(N)}$, Case II is more efficient than Case I; otherwise, Case I is more efficient.

To evaluate relative efficiency, the numerator need be expressed in terms of the denominator. The expected total sum of squares for the case with blocking on random seeds (Case I) is

$$\begin{aligned} ESS_{(Case\ I)} = {} & (I-1)(\sigma^2_\varepsilon + \sigma^2_{AP(N)} + JK\sigma^2_A) \\ & + (J-1)(\sigma^2_\varepsilon + I\sigma^2_{P(N)} + IK\sigma^2_N) \\ & + (I-1)(J-1)(\sigma^2_\varepsilon + \sigma^2_{AP(N)} + K\sigma^2_{AN}) \\ & + J(K-1)(\sigma^2_\varepsilon + I\sigma^2_{P(N)}) \\ & + J(I-1)(K-1)(\sigma^2_\varepsilon + \sigma^2_{AP(N)}) \end{aligned} \tag{4-14}$$

The expected total sum of squares for the randomized design (Case II) is

$$\begin{aligned} ESS_{(Case\ II)} = {} & (I-1)(\sigma^2_\varepsilon + \sigma^2_{P(AN)} + JK\sigma^2_A) \\ & + (J-1)(\sigma^2_\varepsilon + \sigma^2_{P(AN)} + IK\sigma^2_N) \\ & + (I-1)(J-1)(\sigma^2_\varepsilon + \sigma^2_{P(AN)} + K\sigma^2_{AN}) \\ & + IJ(K-1)(\sigma^2_\varepsilon + \sigma^2_{P(AN)}) \end{aligned} \tag{4-15}$$

Since both designs involve the same number of cells, it is reasonable to assume that

$$ESS_{(Case\ I)} = ESS_{(Case\ II)}$$

Setting the right-hand side of Equation (4-14) equal to the right-hand side of Equation (4-15), and simplifying, we have

$$\sigma^2_{P(AN)} = [(I-1)JK\sigma^2_{AP(N)} + I(JK-1)\sigma^2_{P(N)}]/(IJK-1)$$

Thus the relative efficiency with respect to the algorithm effect, $(R.E.)_A$ is greater than 1, i.e., the blocked design is more efficient than the randomized design, if $\frac{(I-1)JK\sigma^2_{AP(N)}}{IJK-1} + \frac{IJ(K-1)\sigma^2_{P(N)}}{IJK-1} > \sigma^2_{AP(N)}$.

This will occur if $\sigma^2_{AP(N)}$ is less than $I\sigma^2_{P(N)}$. Similarly for the N effect, $(R.E.)_N$ is greater than 1, if $\sigma^2_{AP(N)}$ is greater than $\frac{1}{JK}\sigma^2_{P(N)}$.

These results lead to the conclusion that the choice of Case I or Case II depends on the experimenter's purpose. When his principal objective is the comparison of algorithms, he should elect to block on random number seeds (Case I) if it is expected that the variation due to problems will dominate the interaction between algorithms and problems. If nuisance parameter levels are chosen so that the relative difficulty of problems is already encoded in the nuisance levels it is reasonable to assume that problem variance would dominate problem-algorithm interaction. Thus the blocked design is preferred.

On the other hand, if the researcher's goal is to measure the effect of nuisance parameters (as ours will be in Chapter V), the above analysis suggests the fully randomized design of Case II is preferable. Again under the assumption that variation due to problems greatly dominates variation due to problem-algorithm interaction, the Case II approach is relatively more efficient.

CHAPTER V

EMPIRICAL STUDY

In Chapter IV a number of issues regarding the adaption of techniques of statistical experimental design to computational experiments on integer linear programs were presented and analyzed theoretically. In this chapter the analysis of methods for controlled computational experiments on ILP's is extended via empirical studies. These studies have two principal purposes:

1. To provide experience and empirical insight into the issues developed in Chapter IV; and
2. To identify parameters of ILP problems which significantly correlate with the computational complexity of the problem.

Fulfillment of the first objective would help to clarify which design and analysis alternatives a computational experimenter should elect. Fulfillment of the second objective would both suggest which problem parameters may be ignored by experimenters in constructing classes of randomly generated problems and provide information on "What makes integer programs hard?" which is useful in many types of integer programming research.

5.1 Outline of Test Algorithms Used

Two generic LP-based ILP procedures are used in the empirical studies to be reported. They are the Gomory fractional cutting plane algorithm (MIF) and the Land-and-Doig branch and bound algorithm (BB).

Fortran codes for the two procedures were obtained from a recent book by Land and Powell (38), Fortran Codes for Mathematical Programming: Linear, Quadratic, and Discrete. The LP code used by both ILP codes is also adapted from the book. While the algorithms are standard ones, they are briefly outlined below for the convenience of the reader.

Briefly, the LP program can be described as follows. Take a basic point, examine it to see if the conditions for feasibility and optimality of an LP are satisfied. If they are not satisfied, relax one equality condition (so defining an edge*) and move to an adjacent basic point. This unit of calculation is called a "basis change". There are many different rules for choosing the sequence of basic points. The rule used is to satisfy the primal feasibility conditions first by the sequential procedure, i.e., take one infeasibility at a time and perform a basis change iteration until it is satisfied (or until it is established that it is impossible to satisfy it), never violating any primal constraint which is already satisfied. Having achieved feasibility of the primal, perform further basis changes until the dual conditions are also satisfied or until it is discovered that they cannot be satisfied, i.e., that the problem has an unbounded value of the function.

The cutting plane algorithm (MIF) proceeds by solving the linear relaxation LP; checking whether it has integer solution values; if not adding one constraint constructed from each non-integer variable; re-entering and solving the LP from the "now" infeasible basis. The

---

*An edge is the intersection of (n-1) equality constraints.

procedure is repeated until either (i) an integer solution is found at the completion of one of the sequence of LPs, or (ii) the algorithm exceeds the maximum allowed total number of basis changes, or (iii) the coefficients in the generated cuts become too small to give any hope of reaching an integer solution. Any cut constraints which are not explicitly effective are removed, after each LP subproblem is solved. Therefore, it is quite possible for the same cut to be added more than once.

The principle of the branch-and-bound algorithm (BB) is that the convex feasible region of the LP is partitioned into convex subsets and an upper bound (the algorithms are maximizing) on the optimal objective function is obtained for each subset in the partition. If a solution satisfying the integer constraints can be found which has a function value no less than the upper bound on all the subsets, then it is the optimum solution.

The program for BB is based principally on the original Land and Doig method (37). However, it does not branch from the vertex of the branch-and-bound tree with the greatest bound on the objective function (as in the original Land-Doig algorithm). Rather, it proceeds down the main branch until a tail is reached. There are three types of tail:

1. An integer solution.
2. An LP solution with function value at least as low as the best solution discovered so far.
3. An infeasible LP.

Obviously, cases 1 and 3 can only occur on the first tail reached.

When a tail is reached the subsets of the feasible region which have been neglected on the "left" and "right" of the main branch are then investigated. Any which are no better than an integer solution already obtained or are infeasible can be deleted and the level of the tree reduced. But if a level is reached for which either the left or the right bound still offers the possibility of there being a better integer solution within its set, this becomes the main branch. The procedure is repeated.

All the codes use a number of user-set tolerances to control numerical accuracy of the results. Values of the tolerances used in the experiments of this dissertation are shown in Table 7. (See (38) for exact definitions of these tolerances.)

The two ILP codes MIF and BB were selected for use in empirical studies because they are well known and representative of the cutting plane and branch-and-bound classes of LP-based ILP algorithms. However, there is no particular reason to believe they are efficient. Thus the actual solution times reported are probably longer than the one which the best available codes could achieve. For our purposes, however, the solution times will be adequate if they exhibit the same stochastic behavior as times for similar algorithms.

5.2 Summary of the Experiment

As summarized at the beginning of this chapter, the empirical experiments reported in this dissertation had two simultaneous purposes: gaining experience and insight on the design issues developed in Chapter IV and determining which of the problem parameters identified in Chapter III contribute significantly to the difficulty of ILP problems.

Table 7. Accuracy Tolerances Used in Computer Codes

Used in the LP Code

1. Test the feasibility of the basic primal variables = .005
2. Test the feasibility of the slack variables = .005
3. Test the optimality of the dual variables = .005
4. Test the optimality of the objective function row variables = .005
5. Test whether or not a proposed pivot should be regarded as zero = .005
6. Test the relative error of the primal variables of a solution on each constraint = .005
7. Test the relative error of the dual variables of a solution on each variable = .005
8. Test the size of a proposed pivot during a re-inversion of the basis = .05

Used in the MIF Code and the BB Code

Test whether or not a variable has an integer value = .001

To fulfill these objectives simultaneously, an experiment was conducted which observed the performance of the two generic ILP algorithms presented in Section 5.1 on problems generated with a wide range of settings of the problem parameters. Results of the experiment were then analyzed via the techniques of Chapter IV. Thus the experiment had as its primary focus the identification of which problem parameters contribute significantly to ILP problem difficulty, but the use of analysis procedures developed in Chapter IV permitted the simultaneous gaining of insight and experience with those procedures.

Recall that the analysis of Section 4.4 suggested that if an experimenter's purpose was the analysis of the effect of problem or nuisance parameters, no advantage would probably be gained by blocking on random number seeds, i.e., using the same randomly generated problems on all solution algorithms. Moreover, it was anticipated that the problem parameters which correlate with problem difficulty would differ between cutting plane and branch-and-bound algorithms. Thus two separate experiments were actually run--one a 2^7 factorial experiment on the cutting plane procedure MIP, and the other a 2^7 factorial experiment on the branch-and-bound procedure BB. The seven problem or nuisance factors used in each case are listed in Table 8. (See Chapter III for exact definitions.)

5.2.1 Selection of Low and High Parameter Values

As reported in Chapter III, the parametric problem generator developed in this dissertation research can control, at least partially, each of these problem parameters. A preliminary study was made to find what range of these parameters generated problems hard enough to be

Table 8. Problem or Nuisance Parameters Varied in Experiments

1. Distance index between LP and IP optimality.
2. Degree of dual degeneracy in the LP optimal basis.
3. Degree of primal degeneracy in the LP optimal basis.
4. Density of nonzero entries in the constraint matrix.
5. Determinant of LP optimal basis.
6. Total number of integer variables.
7. Total number of constraints.

interesting in the main experiment. The criterion used to characterize such problems was that average solution time for the two procedures used should be in the neighborhood of 30 CPU seconds on Georgia Tech's Univac 1108. Some difficulty was encountered in choosing the low and high levels for each parameter because several parameters interact with one another. As discussed in Chapter III, to effectively control the density of nonzero entries in the constraint matrix of the problem it is better to have the number of integer variables at least twice the number of constraints. Suppose we set the low and high levels of the number of constraints 10 and 25, respectively. This implies that it would be better to set both the low and high levels of number of integer variables greater than 50. ILP problems of this size are often extremely hard to solve by the procedures used.

A similar problem was encountered with the dual and primal degeneracy. It can be argued that these parameters do affect problem difficulty, but no measures have been proposed to differentiate the

degree of the effect. If the absolute number of degenerate values in the optimal LP solution were used as a measure, then say 8 out of 25 variables degenerate would be equated in difficulty with 8 out of 10. It was therefore decided to use the percent degenerate to measure primal and dual degeneracy. The high and low levels used for these and the other parameters in the empirical study are summarized in Table 9.

Table 9. High and Low Level of Parameters Used in Experiments

Problem Parameter	Low Level	High Level
Distance Index	0	1
Dual Degeneracy	0%	20%
Primal Degeneracy	0%	40%
Density	.20	.40
Determinant	256 ($=2^6$)	65,536 ($=2^{12}$)
Number of Integer Variables	30	40
Number of Constraints	5	15

5.2.2 Treatment of Time Limits

A 240-second limit on CPU time was set for solving all test problems. If any problem was not solved within the limit, computations stopped and the observation was recorded as censored with the time limit being used as the censor point. This scheme worked quite satisfactorily with the BB procedure; but, the MIF procedure sometimes failed before the time limit was reached. Such failures arose in situations where the

values of the cut coefficients became so large that no useful cutting plane could be generated. Such observations were also treated as censored, but the times where failures occurred were recorded as the censor points instead of the maximum time limit.

5.2.3 Response Statistics Measured

Enough response statistics were collected from the experiments to provide any of the response variables discussed in Section 4.1. For both the algorithms, the following data were recorded:

1. Time for initial LP.
2. Number of iterations for initial LP.
3. Total integer time.
4. Sum of time for restart LP's.
5. Sum of iterations for restart LP's.
6. Number of subproblems solved.

All times were in CPU seconds on the Univac 1108. The quantities are defined in Section 4.1.

5.2.4 Replication

From preliminary observations, it was anticipated that large variations would be observed within experimental cells and that a rather high percentage of the experimental data would be censored. As a result, two replicates of the 2^7 experiment were run for each ILP algorithm. The data collected are shown in detail in Appendices B and C.

5.3 Analysis of Experimental Data

5.3.1 Choice of a Response Variable

The choice of response variable for a designed experiment is

usually made at the experiment planning stage. However, since one part of this research is analysis of which response variables are appropriate in computational experiments, several different response variables were analyzed in the experiments on MIP and BB.

In Table 10 the most important of the measures developed in Section 4.1 are compared in a correlation analysis. The table shows correlation coefficients of solution time vs. equivalent iterations from LP to IP optimality and solution time vs. LP subproblems. All statistics are for operations after the initial LP only.

Table 10. Correlation Coefficients of Solution Time vs. Equivalent Iterations from LP to IP Optimality and Solution Time vs. LP Subproblems

	MIP Integer Solution Time	BB Integer Solution Time
Equivalent iterations from LP to IP Optimality	.874	.968
LP Subproblems	.795	.836

As the table indicates, both the number of LP equivalent iterations and the number of LP subproblems are closely correlated with solution time. However, the correlation is higher for LP equivalent iterations than for LP subproblems and higher for the BB code than for the MIP code. Since the correlations are so high integer solution time was used as the response variable in all analyses of this chapter.

The results in Table 10 appear to lead to the general conclusion that no important differences in algorithm rankings would arise if one

of these measures were substituted for another as a response variable. However, some caution is warranted in drawing any such conclusion because of the nature of the ILP codes used in this experiment. Especially in the case of the BB code, very little post-optimality analysis is performed after each LP subproblem. Thus the high correlations of Table 10 could be expected. Somewhat different results might arise if a code were investigated which used time-consuming bound calculation routine, in an effort to reduce the number of LP subproblems.

5.3.2 Occurrence of Censoring

A summary of the occurrence of censored observations in the experimental data is given in Table 11. The "Partially censored cells" mentioned in the table are cells where only one of the two replicates is censored and "Completely censored cells" are cells with both replicates censored.

Table 11. Occurrence of Censored Observations

Type of Observation	Cutting Plane (MIF)	Branch and Bound (BB)
Censored in partially censored cells	13 (5.1%)	28 (10.9%)
Censored in completely censored cells	40 (15.6%)	14 (5.5%)
Sub-total censored	53 (20.7%)	42 (16.4%)
Uncensored	203 (79.3%)	214 (83.6%)
Total	256 (100%)	256 (100%)

From that table it is clear that censoring was a more serious problem with the cutting plane than with the branch and bound algorithm. Note first that the proportion of observations censored was greater for MIP. However, the more serious problem is that the cutting plane algorithm tends to have a larger proportion of completely censored cells; whereas, the branch and bound algorithm tends to have more partial censoring. As was discussed in Section 4.2.2, the occurrence of cells with no uncensored data (i.e., completely censoring) particularly complicates the estimation of censored values.

It is also important to note that the total number of censored observations was in the 15-20% range for both procedures. Thus the approximate F-tests developed by analogy in Section 4.3.2 could be expected to give fairly satisfactory results when estimates of censored values are used in place of true observations.

5.3.3 Preliminary Analysis of Data Transformations

As outlined in Sections 4.2 and 4.3, the assumptions of normal and identically distributed (equal variance) residuals are important elements of the statistical models underlying both the analysis of variance and the Sampford and Taylor procedure for estimating censored experimental values. Figures 3 and 4 show the scatter diagrams of cell mean vs. cell variance of the residuals observed when an ANOVA model was fitted to experimental results for the BB and MIF algorithms. For purposes of these figures, censore points were used in place of censored observations. It is evident in the figures that there was a trend between cell means and cell variances, i.e., larger cell means tended to have larger cell variances. Thus it was concluded that some

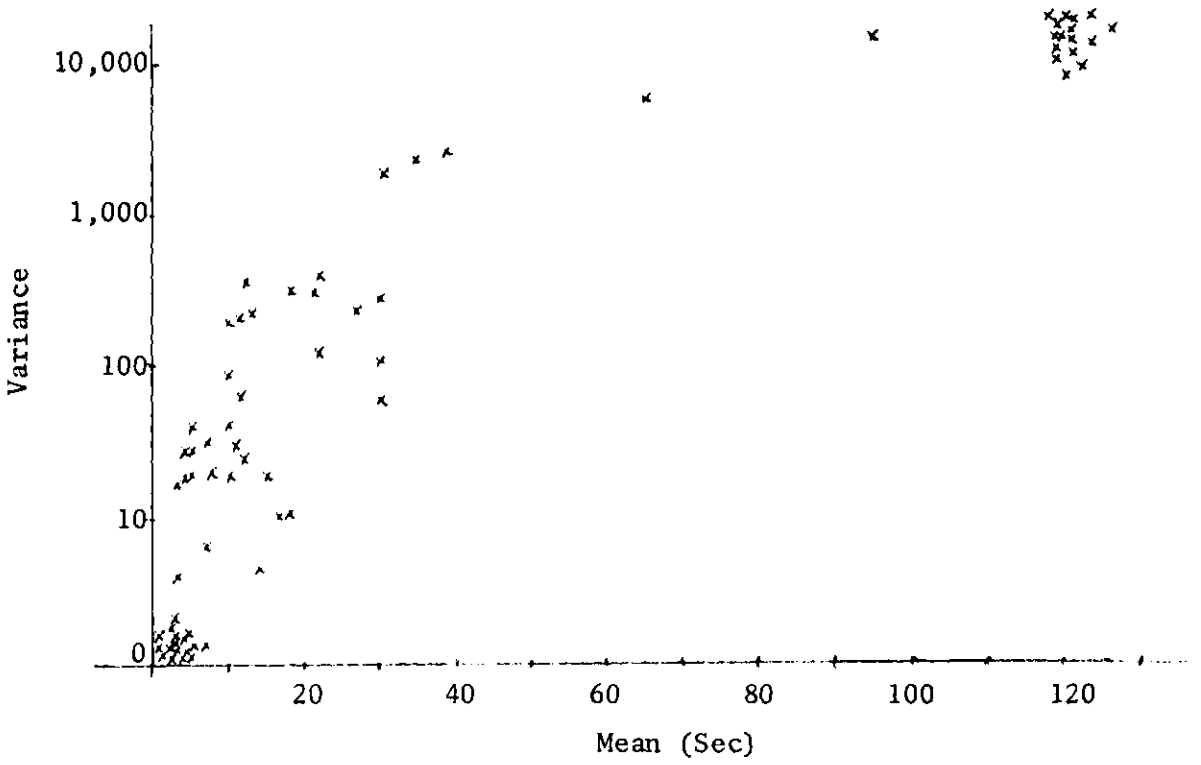

Figure 3. Scatter Diagram of Mean vs. Variance (Untransformed BB Observations)

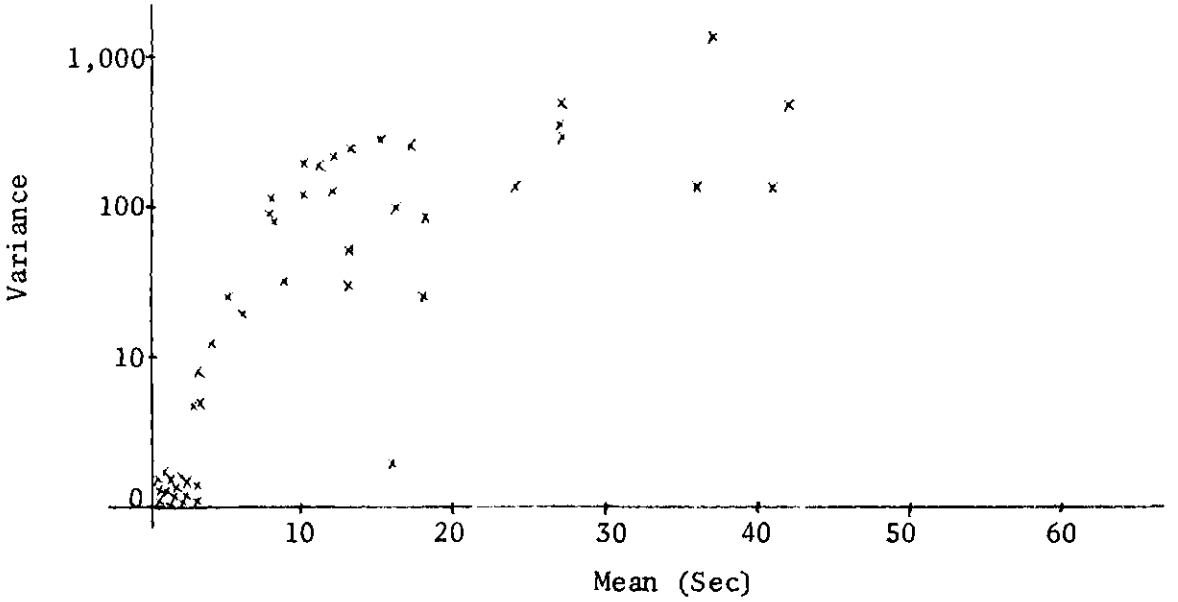

Figure 4. Scatter Diagram of Mean vs. Variance (Untransformed MIF Observations)

form of data transformation was required to meet the assumptions of the ANOVA and Sampford-Taylor procedures.

Note, however, that complete application of the Box and Cox procedure presented in Section 4.3 for choosing an appropriate transformation requires the availability of all experimental observations. At the same time application of the Sampford and Taylor procedure for filling in censored values requires data to already be transformed so that normality and equal variance assumptions are satisfied. This dilemma was resolved by a two-stage process. In the first stage, censor points were used in place of censored observations and the Box and Cox procedure was applied to select a transformation. Uncensored data were then used under this transformation to estimate censored observations via the Sampford and Taylor procedure. Finally, the Box and Cox procedure was reapplied, to select a transformation--this time using the values from the Sampford and Taylor procedure in lieu of the censored observations.

In the Box and Cox procedure, the best value of λ in the transformation may be determined by plotting the maximized log likelihood $L_{max}(\lambda)$ against λ for a trial series of values. As outlined above, the procedure described in Section 4.3 was programmed for this purpose and applied first to a preliminary data set consisting of uncensored values and censor points as estimates of censored values. A plot of $L_{max}(\lambda)$ vs. λ for the branch-and-bound algorithm (BB) is shown in Figure 5. The figure indicates an optimal value of approximately $\hat{\lambda} = .1$, and shows that the 99 percent confidence interval is approximated from -.01 to -.13. Similarly, a plot of $L_{max}(\lambda)$ against λ for the cutting

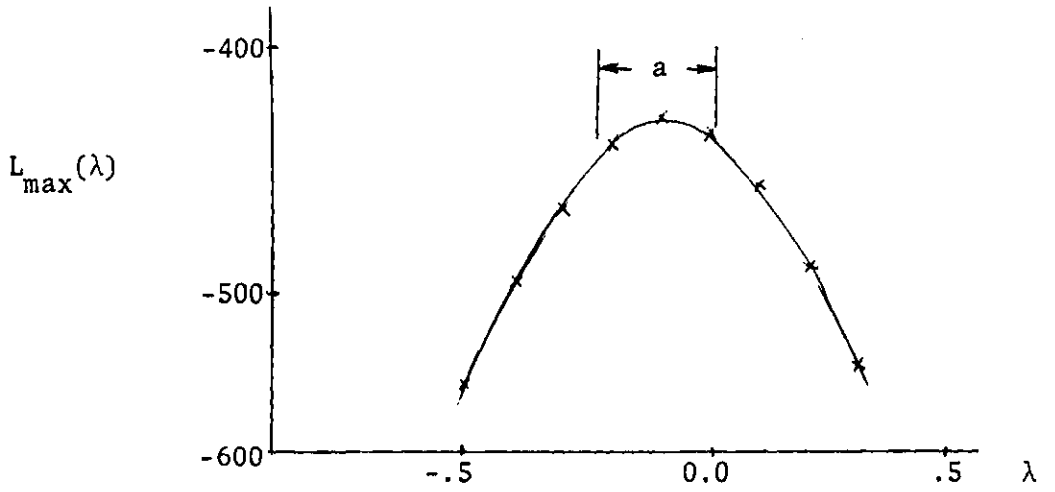

Figure 5. $L_{max}(\lambda)$ Against λ for BB

a: 99% Confidence Interval

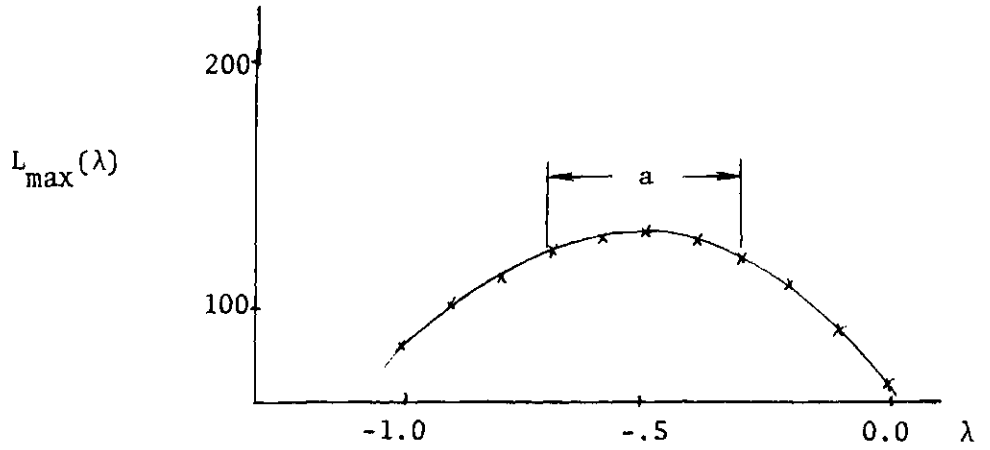

Figure 6. $L_{max}(\lambda)$ Against λ for MIF

plane algorithm (MIP) is shown in Figure 6. The figure indicates an optimal value of about $\hat{\lambda} = -.5$ and an approximate 99 percent confidence interval for λ extending from about -.35 to -.65.

5.3.4 Estimation of Censored Values

Section 4.2 detailed the Sampford and Taylor, maximum likelihood method for estimating censored values in the case of randomized block experiments. When a replicated factorial design has all values censored in any cell it was suggested in Section 4.2.2 that the procedure applied separately to each replicate because maximum lieklihood estimates of missing values are the (nonexistent) cell mean. Since a number of cells were completely censored in our experiment this latter approach was used in estimating censored values. Initially, the censoring procedure was applied to uncensored data transformed with $\lambda = -.1$ for the branch and bound algorithm and $\lambda = -.5$ for the cutting plane algorithm, i.e., the values chosen by the preliminary analysis of the previous section. It was found that the estimated values for censored data produced by this technique were unreasonably large and widely separated from the uncensored values. Thus several different values of λ were tried for each algorithm. It was observed that the estimated values for censored data become smaller as λ moves closer to zero from the minus side. It appeared $\lambda = -.05$ gave reasonable results for the BB algorithm. Since the transformation $\hat{\lambda} = -.1$ was only approximate and based on censor points in lieu of censored observations, the $\hat{\lambda} = -.05$ value was selected. Similarly for the cutting plane algorithm MIF, several values of λ were attempted. A best value of λ appeared to be $\hat{\lambda} = -.1$.

5.3.5 Final Analysis of Data Transformations

As outlined above, the final step in the analysis was to apply the Box-Cox procedure a second time to see what transformations were appropriate after estimate of the censored data were available. Figures 7 and 8 plot the revised $L_{max}(\lambda)$ versus λ for the BB and MIF procedures respectively. The 99 percent confidence interval for the cutting plane data is (-.54, -.49) with the maximum at $\hat{\lambda} = -.52$, and the corresponding interval for the branch-and-bound algorithm is (-.13, -.09) with the maximum at $\hat{\lambda} = -.11$.

Note that these values are almost the same as those in Section 5.3.3. When the percent of censored values is in the range given in Table 11, this seems to suggest that the preliminary transformation procedure of Section 5.3.3 is adequate. This additional effort of a two step choice of λ, with censored value estimation between steps, appears to have produced little change in the results.

Figures 9 and 10 show the histograms of residuals for both the branch-and-bound and cutting plane algorithms. Both figures appeared unimodal and symmetric, but the Chi-square test indicated that both histograms were not normally distributed. The Chi-square statistic calculated for BB was 43.74 and 140.81 for MIF. Both were rejected at the .1% significance level.

The scatter diagrams of cell means and cell variances for transformed observations along with estimated censored data are shown in Figures 11 and 12. Since Figures 11 and 12 appeared scattered around a flat straight line, it was suggested that cell variances were rather uniform. Two replicates of each algorithm, however, were not enough

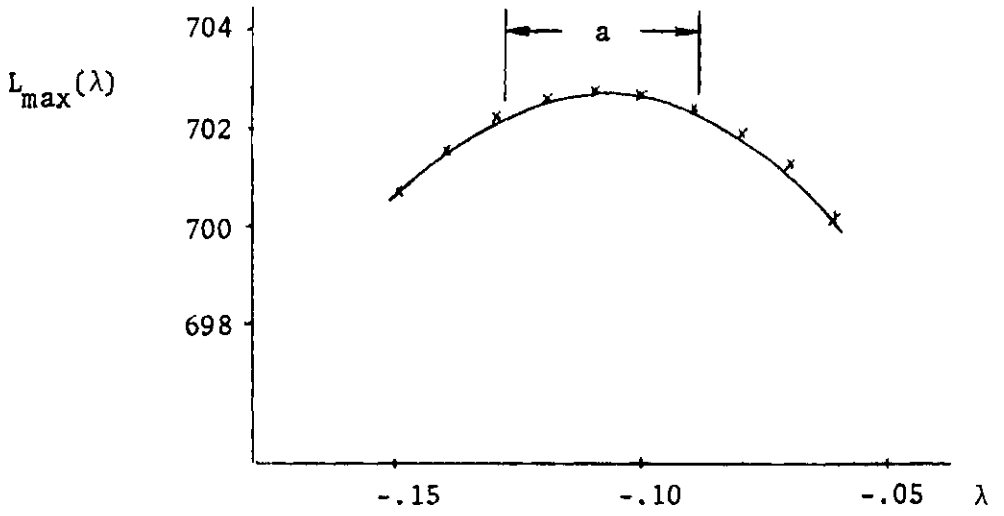

Figure 7. $L_{max}(\lambda)$ Against λ for BB

a: 99% Confidence Interval

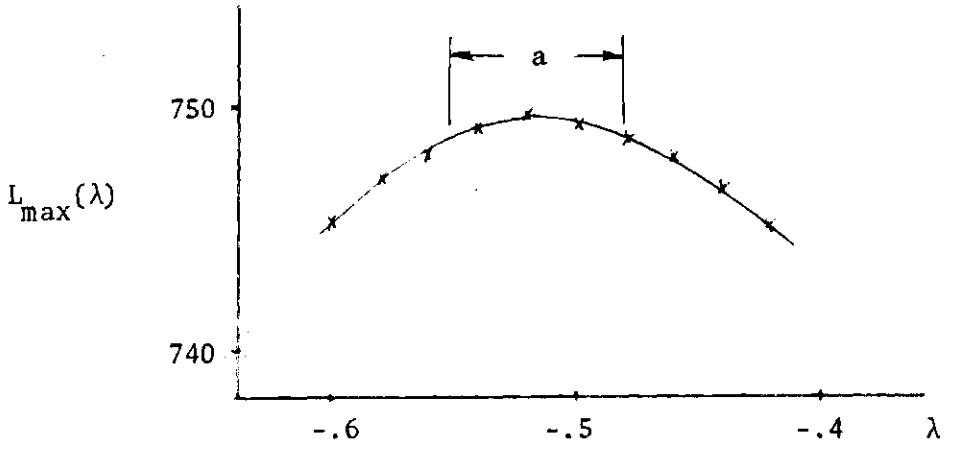

Figure 8. $L_{max}(\lambda)$ Against λ for MIF

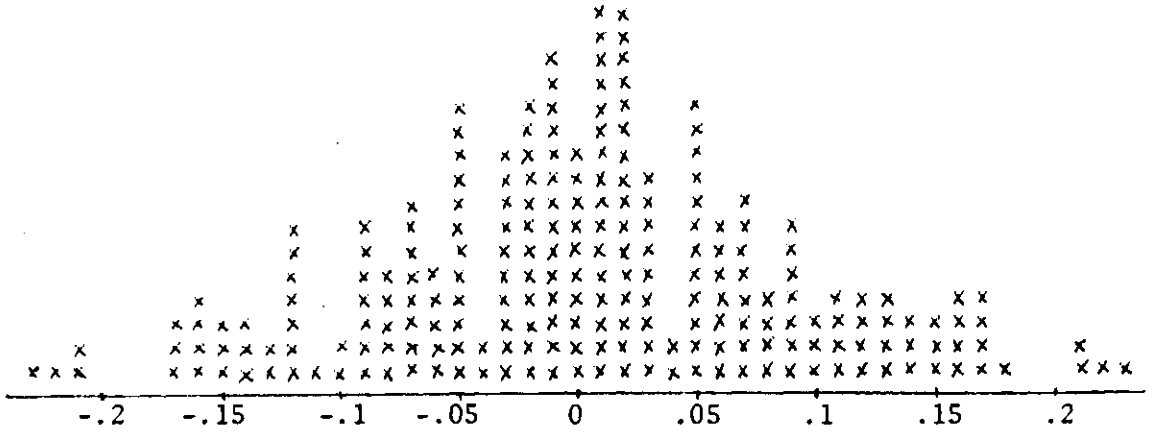

Figure 9. Histogram of Error Residuals for BB (Transformed Observations with Estimated Censor Points)

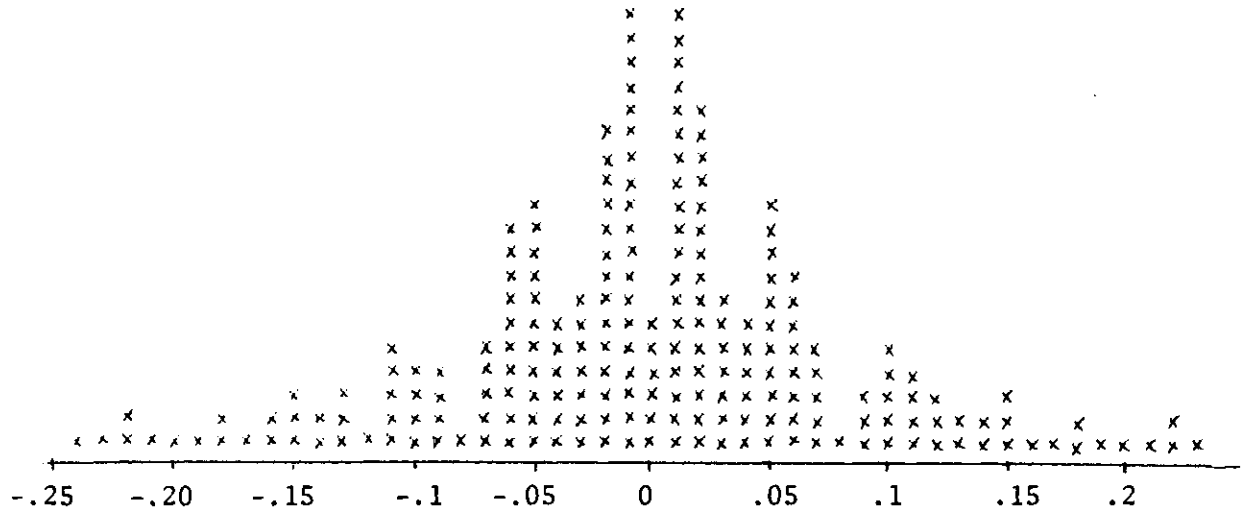

Figure 10. Histogram of Error Residuals for MIF (Transformed Observations with Estimated Censor Points)

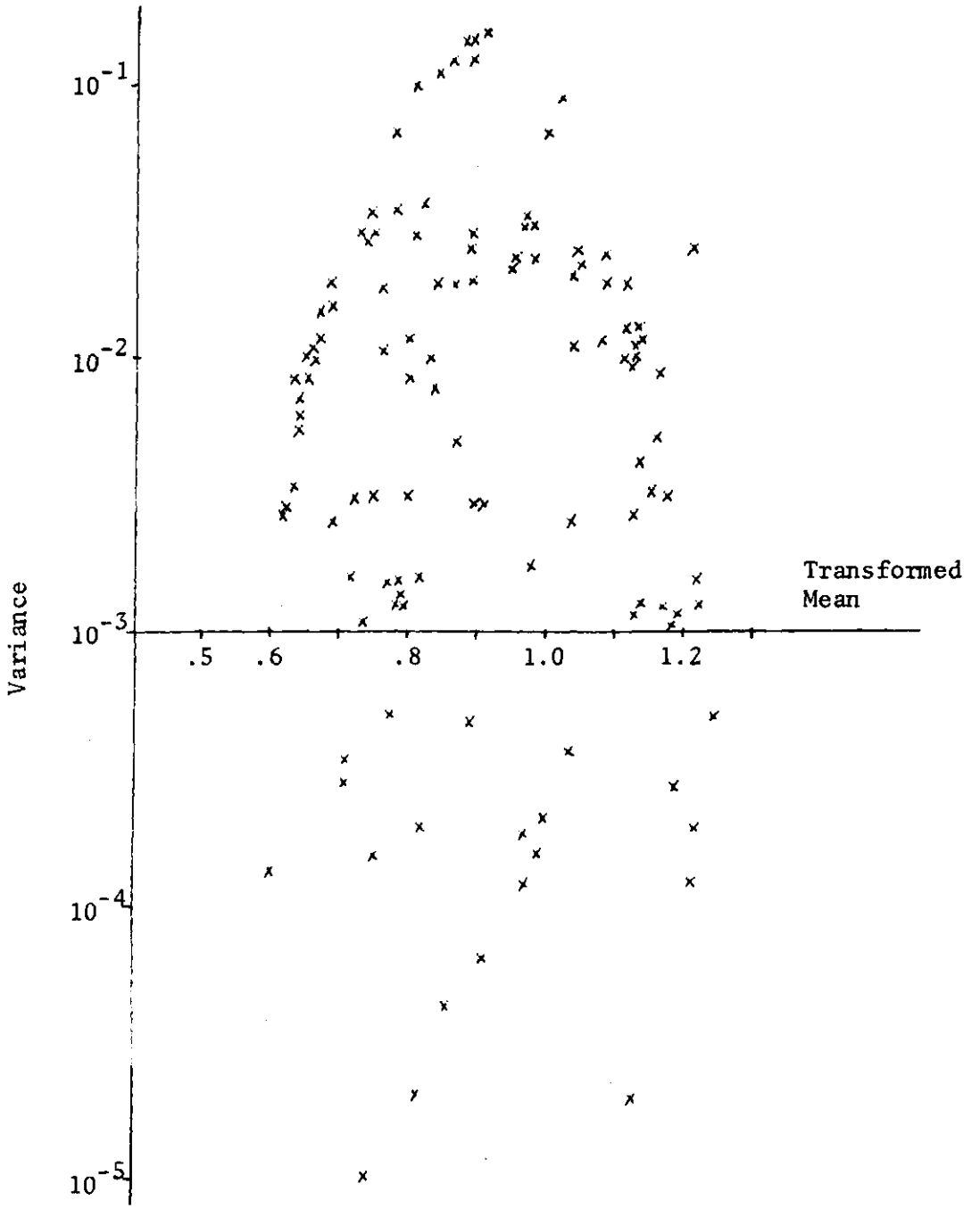

Figure 11. Scatter Diagram for Transformed Branch-and-Bound Observations with Estimated Censor Points

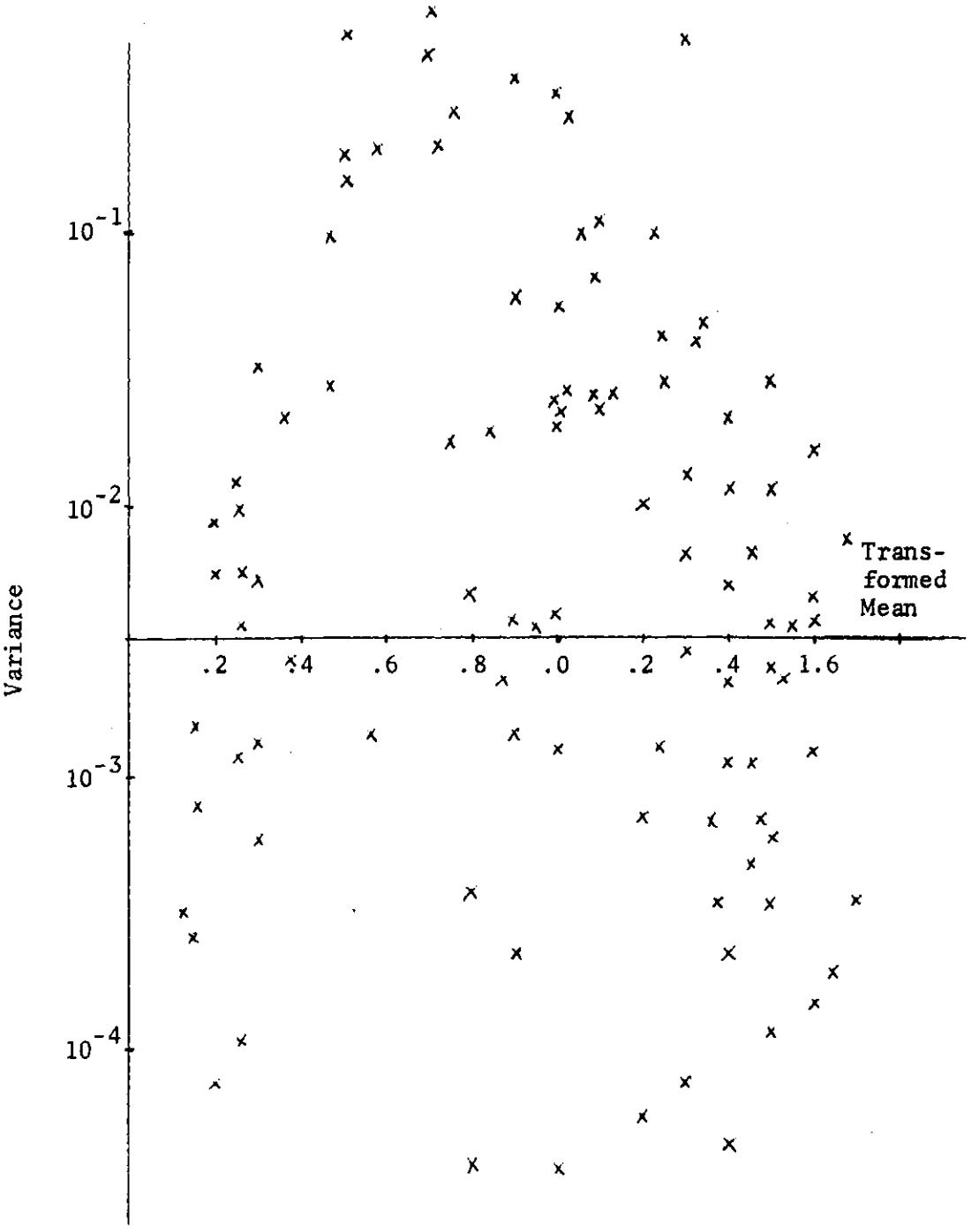

Figure 12. Scatter Diagram for Transformed Cutting Plane Observations with Estimated Censor Points

to further justify equality of cell variances by statistical test. In order to obtain some more information about the equality of cell variances under the transformation, data were pooled. Tables 12 and 13 show the sum of squares for each main effect in the two experiments under the final transformation. From the sum of squares, it appeared that the determinant of LP basis was relatively insignificant for the branch-and-bound algorithm. Similarly, the primal and dual degeneracy effects appeared very unimportant for the cutting plane results. Thus data for BB were pooled by collapsing two replicates of a 2^7 experiment into four replicates of 2^6 experiment, omitting the apparently insignificant determinant effect. Similarly, data for MIF were collapsed into eight replicates of 2^5 experiment.

The computed Bartlett statistic for equality of variances in the branch and bound case was 135.98, which leads to rejection of the equality hypothesis at $\alpha = .1\%$. However, the computed Levene statistic was .553 and $F_{63,192}(.75) = 1.15$. Therefore, the hypothesis of equality of variances failed to be rejected at $1 - \alpha = 75\%$. This discrepancy was attributed to the departure from the normality assumption already mentioned.

The cutting plane algorithm experienced a similar discrepancy. When the residual variances in the collapsed experiment were analyzed the computed Bartlett statistic was 175.24, which again results in rejecting the hypothesis of equality at the .1% level. The Levene statistic was 1.275 and $F_{31,224}(.75) = 1.23$ so that the equality hypothesis is not rejected at the $1 - \alpha = 75\%$ level.

While these tests of the validity of the normality and equal

Table 12. Sum of Squares for BB

	Source of Variance	Sum of Squares
A	(Replication)	.0059
B	(Distance Index)	2.98627
C	(Dual Degeneracy)	.28494
D	(Primal Degeneracy)	.93132
E	(Density)	1.12245
F	(Determinant)	.05402
G	(No. of Variables)	.46171
H	(No. of Constraints)	.16896

Table 13. Sum of Squares for MIF

	Source of Variance	Sum of Squares
A	(Replication)	.53888
B	(Distance Index)	25.31080
C	(Dual Degeneracy)	.17724
D	(Primal Degeneracy)	.01599
E	(Density)	12.24069
F	(Determinant)	.85368
G	(No. of Variables)	6.24365
H	(No. of Constraints)	.99454

variance assumptions of the analysis of variance procedure are not entirely satisfying, they do appear to justify the application of ANOVA to the transformed data. Residuals under the transformations are apparently non-normal, but they are unimodal and symmetric. Many studies have shown that the ANOVA is robust to normality as long as residuals have at least these two properties. Moreover the above analysis of the more crucial equality of variance assumption appears to show this assumption is well approximated by the transformed data. When a test is applied which is insensitive to the non-normality, equality of variance is not rejected at $1 - \alpha = .75$.

5.4 Final Analysis of Variance and Empirical Results

As outlined in Section 4.2.2 applying the Sampford and Taylor method to individual blocks of a factorial design has the effect of turning replicates into blocks. Thus the final analysis of the transformed data was treated as a 2^7 blocked factorial experimental design with two blocks. The analysis of variance program in the BMD (Biomedical) computer package was used to do the computations. Complete analysis of variance tables, one for the cutting plane algorithm and the one for the branch-and-bound algorithm, are shown in Appendices D and E, respectively.

Following the usual practice, interaction between blocks (replicates) and all treatment effects were assumed zero and treated as residual error. This would normally yield 126 degrees of freedom for the error mean square. However, 110 (BB) and 106 (MIF) were used instead of 126 to adjust for the estimation of censored values as

suggested in the Sampford and Taylor procedure.

The theoretical F-statistics, $F_{1,110}(.99)$ and $F_{1,106}(.99)$ are approximately 6.90; $F_{1,110}(.95)$ and $F_{1,106}(.95)$ are approximately 3.93. Main effects and interactions whose F-ratios are greater than 3.93 are listed in Table 14 from the branch-and-bound algorithm and Table 15 for the cutting plane algorithm. Tables 16 and 17 give the estimates of problem parameter effects when moving from the low level to the high level of each effect. As the original two sets of data were transformed by using a reciprocal transformation, negative values in Tables 16 and 17 indicate that mean solution times are higher at the high level settings, and conversely larger positive values indicate that mean solution times are higher at the low level settings.

Recall that the data analysis and the F-test proposed in this research is only approximate. It must be stressed that a value near an operative significance level of the F-distribution should be accepted as significant at that level only with reservations.

5.4.1 Main Effects and Interactions for BB

Distance index, as shown in Table 14, is significant. The estimate of the distance index effect, presented in Table 16, at the low and high level indicates that ILP problems would become harder for BB when the distance between LP optimality and IP optimality is wider. This result was confirmed by printouts of the BB code. All integer feasible solutions in the path to IP optimality were reported in the computer output. It was observed that high distance-index test problems contained more integer feasible solutions than low distance-index problems did. Thus the algorithm does appear to have to move farther

Table 14. Analysis of Variance Table for Branch and Bound Results. [Only main and significant effects included]

Source of Variance	df	Mean Square	F Ratio
A (Replication)	1	.00059	.0212
B (Distance Index)	1	2.98627	92.374**
C (Dual Degeneracy)	1	.28494	8.659**
D (Primal Degeneracy)	1	.93132	28.811*
E (Density)	1	1.12245	29.094**
F (Determinant)	1	.05402	1.669
G (No. of Variables)	1	.46171	11.953**
H (No. of Constraints)	1	.16896	5.226*
B x G	1	.16418	4.914*
C x D	1	.18469	5.713*
C x F	1	.19363	5.018*
D x H	1	.20012	6.189*
G x H	1	1.19056	36.825**
B x C x H	1	.14031	4.265*
B x G x H	1	.41334	12.243**
C x E x H	1	.17551	5.373*
Residual	110	.03233	

*Significant at 5%
**Significant at 1%

Table 15. Analysis of Variance Table for Cutting Plane Results. [Only main and significant effects included]

Source of Variance	df	Mean Square	F Ratio
A (Replication)	1	.53888	8.767**
B (Distance Index)	1	25.31080	361.089**
C (Dual Degeneracy)	1	.17724	1.945
D (Primal Degeneracy)	1	.01599	.187
E (Density)	1	12.24069	164.495**
F (Determinant)	1	.85368	11.842**
G (No. of Variables)	1	6.24365	85.643**
H (No. of Constraints)	1	.99454	13.641**
B x C	1	.86511	11.864**
B x E	1	1.08711	14.899**
B x F	1	.45515	6.242*
B x H	1	3.63337	49.827**
E x G	1	.33648	4.616*
F x H	1	.38864	4.552*
G x H	1	2.78162	38.148**
B x C x H	1	.51825	7.038**
B x F x H	1	.45512	5.328*
B x G x H	1	2.57960	35.392**
Residual Error	106	.07292	

*Significant at 5%
**Significant at 1%

Table 16. Estimates of Main Effects for BB.
[Using transformed data]

Effects	Estimate
A. Replication	.0030
B. Distance Index	-.2160
C. Dual Degeneracy	-.0667
D. Primal Degeneracy	.1206
E. Density	-.1324
F. Determinant	-.0290
G. No. of Variables	-.0849
H. No. of Constraints	.0513

Table 17. Estimates of Main Effects for MIF.
[Using transformed data]

Effects	Estimate
A. Replication	.0924
B. Distance Index	-.6288
C. Dual Degeneracy	.0526
D. Primal Degeneracy	-.0158
E. Density	-.4373
F. Determinant	-.1155
G. No. of Variables	-.3123
H. No. of Constraints	.1247

from the LP optimal solution to find an IP optimal solution when distance index is high.

Dual degeneracy is significant. Table 16 indicates that ILP problems would require more time for BB to solve if the degree of dual degeneracy in the optimal LP basis is higher. As noted in Chapter III, ILP problems with high degree of dual degeneracy contain a number of optimal LP bases. Thus if the algorithm fixed on value, another optimal LP basis may arise which has no decrease in objective function value. It could waste a great deal of time to search through all such solutions before a change in the bound occurs.

Primal degeneracy is also significant. It can be seen in Table 16 that the lower the degree of primal degeneracy in the optimal LP basis, the longer it apparently takes for BB to solve ILP problems. Based on knowledge of linear programming, we would expect more time to be required in order to reach LP optimal solutions when the degree of primal degeneracy is higher. In the ILP case, however, a higher degree of primal degeneracy in the LP optimal basis implies more variables of LP optimal solution in integral form, namely, zero. In fact, the degree of primal degeneracy is also a measure of fraction of LP optimal solution in integral form. This consequence for the LP optimal solution apparently dominates any increases in LP time.

Density is significant. In Table 16, it is indicated that high density of nonzero entries in the constraint matrix would make ILP problems harder. This effect is consistent with other studies reported in the literature. The reasoning for this effect may be given in two ways. In each LP subproblem, it would take more time to determine which

element to pivot if density of nonzero entries is high. The other reasoning is that more integer feasible solutions are present around the LP optimal solution when density is high because there are numerous combinations of nonbasic variables which will make basic values integer. Therefore it is likely that the algorithm would need to search through many integer feasible solutions not containing an optimal solution before strong bounds are obtained. The effect may thus be quite similar to the dual degeneracy effect.

As indicated in Table 16, determinant of LP optimal basis is rather insignificant. This seems to refute the idea suggested by Jeroslow that the determinant should be used to indicate the hardness of ILP problems, but is not really inconsistent with the literature. All theoretical work on determinant basis is in the context of cutting plane or related algorithms.

It is a common empirical result in the literature that the computational complexity of ILP problems largely depends upon the number of integer variables--the more the harder. The estimate of this effect, as presented in Table 16, confirms this kind of statement.

The number of constraints is significant as shown in Table 14. The estimate of the effect of the number of constraints in Table 16 indicates that ILP problems with more constraints tend to be easier to solve. While ILP problems with more constraints would require more time to solve each LP subproblem, it also means that these problems contain more restricted feasible region. The latter implies fewer potential LP subproblems needed to be solved. This observation corresponds to the recent findings by Geoffrion and Graves (18) among others (as

discussed in Chapter II) that--contrary to the LP case--ILP problems with more constraints are easier.

As was mentioned earlier, only significant interactions are listed in Table 14. Out of eight such interactions, two interactions are highly significant, namely G x H and B x G x H, where B (Distance Index), G (No. of Variables), and H (No. of Constraints). Since a BG interaction is also moderately significant, the BGH interaction is probably a secondary impact of the B x G and G x H interactions. A graphic illustration of the particularly strong interaction G x H is given in Figure 13. The interaction is apparently due to nonlinear growth of solution time with respect to number of integer variables. This result agrees with empirical work reported in Geoffrion (20) and elsewhere.

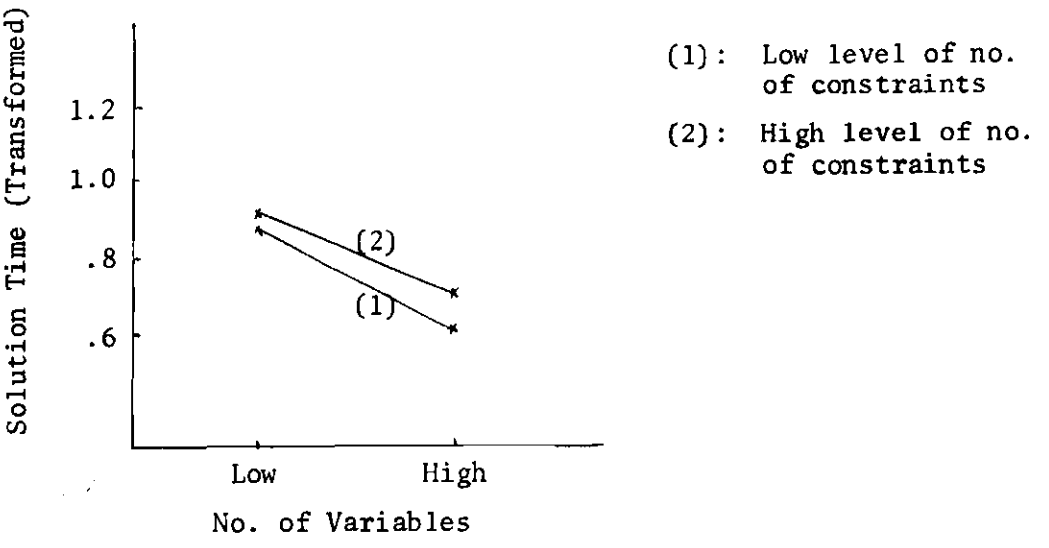

Figure 13. Graph of Number of Variables vs. Number of Constraints Interaction in Branch and Bound Results

Other significant interactions seem due to similar nonlinear behavior of ILP problems. A check of the implications of the C x F (dual degeneracy vs. determinant) interaction was performed to see whether it might be masking effects of determinant. The check revealed no contradiction of the earlier conclusion that determinant is insignificant.

5.4.2 Main Effects and Interactions for MIF

Contrary to the experience with BB, the replication effect caused by estimation of censored values was significant for MIF. This phenomenon is probably due to the greater problem with censoring in the MIF case already noted in Section 5.3.2.

Similar to BB, distance index is highly significant. The estimate of this effect, as presented in Table 17, indicates that ILP problems tend to be harder to solve if the gap between LP optimality and IP optimality is large. A likely reason for this is that more cuts are needed to traverse the relatively large distance to IP optimality when distance index is high.

Unlike BB, the degrees of primal and dual degeneracy in the LP relaxation of ILP problems are rather insignificant. As discussed earlier, a high degree of primal degeneracy implies large fraction of the LP optimal solution in integral form and dual degeneracy implies the need to evaluate many LP subproblems before bounds improve. Since the solution strategy for MIF does not make much use of integrality in the LP solution and need not search poor LP solutions, the degree of degeneracy apparently does not effect solution times of ILP problems solved by MIF.

Another cause of the insignificance of degeneracy may be that MIF generates one cut for each fraction variable. Thus lack of integrality in x_B may actually cause more of the necessary cuts to be generated early. Similarly most of the alternative optima implied by dual degeneracy may be cut off simultaneously by the several cuts.

As with BB, Table 15 indicates that density of nonzero entries in the constraint matrix is significant--the higher the density, the harder the ILP problems. Besides the fact that each LP subproblem would require a longer time when density is high, the high frequency of integer feasible point near IP optimality, noted earlier in the BB case, may impede generation of deeper cuts. Thus high density problems may require adding a great number of cuts before reaching IP optimality.

The determinant of the optimal LP basis is highly significant as expected from the findings of Jeroslow and others. The estimates of the determinant effect in Table 17 indicates that the larger the determinant, the harder the ILP problems. However, the results do not confirm Jeroslow's claim that the determinant is a more important factor than such factors as the number of variables or the number of constraints. As shown in Table 15, the F-ratio of the determinant is no greater than those of the other two factors. But in comparing F-ratios it should be noted that the "high" level of the determinant effect was 65,636 ($=2^{12}$). This is not a particularly large number, but an excessively large number was not used because of the numerical difficulties for the computer such small fractions would imply.

Similar to BB, the number of variables is a highly significant parameter. All results confirm the notion that ILP problems with

large number of variables are harder to solve.

Table 15 indicates the effect of the number of constraints was more significant than for BB. The estimates of this effect in Table 17 indicates that more of constraints make ILP problems easier as is the case with BB. Again the observation of Geoffrion (18) and others are confirmed.

It can be seen in Table 15 that, as compared to BB, more interactions of MIF are significant, and the distance index effect tends to interact with many other effects. The distance index interactions appear to be nonlinear behavior caused by other effects when distance index is high. For example, Figure 14 shows the distance index vs. number of constraints interaction. As is in the BB case, the number of variables also interacts significantly with some other effects. This again indicates the nonlinear effect of the number of variables.

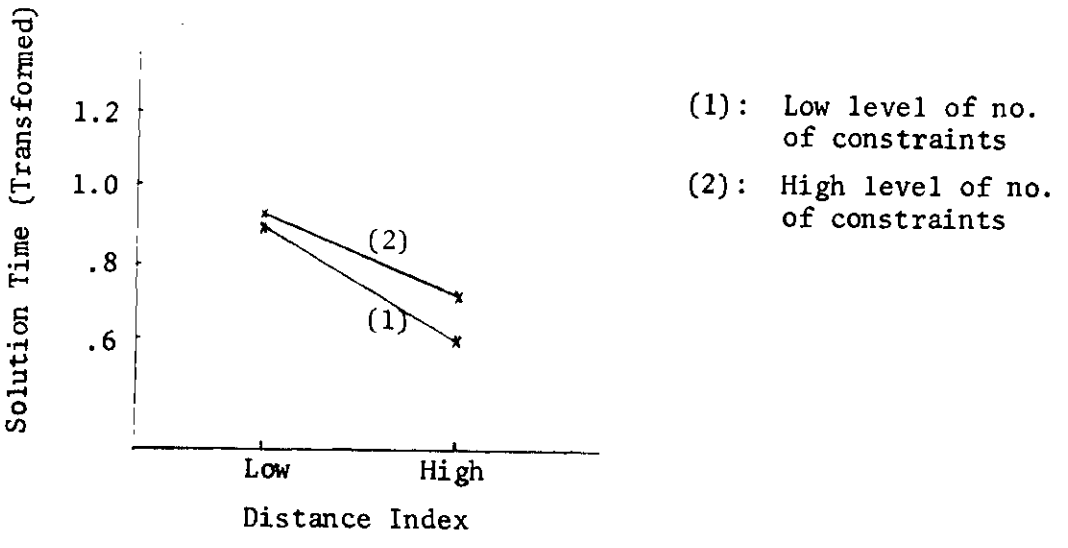

Figure 14. Graph of Distance Index vs. Number of Constraints Interaction in Cutting Plane Results

As in the case of BB, checks were made to assume that the significant interactions had not produced misleading results for the main effects. These checks confirmed the significances in Table 15.

5.4.3 Summary

A number of conclusions can be drawn from this empirical study. Interestingly, the statistical results seem to confirm the observations about the computational complexity of ILP problems made individually by other studies. Jeroslow, for example, advocated that the determinant of the LP optimal basis as the most important factor. While those results do not completely confirm his view, the results do indicate that the determinant effect is highly significant for the cutting plane algorithm. Geoffrion observed that solution times of ILP problems by BB grow exponentially with the number of integer variables; the results seem to confirm that the effect of number of variables is highly significant and nonlinear. As discussed in Chapter II, Geoffrion and Graves, among others, discovered that formulations with more constraints may make ILP problems easier to solve; their finding seems confirmed by our results.

While it might have been expected that different solution strategies would encounter different problems with nuisance parameters, the results provide statistical evidence showing such suspicion is justified. According to the above results, the determinant effect is highly significant for MIF, but not for BB; and the effects of primal and dual degeneracy are just the reverse.

Finally the results indicate that the distance index yields perhaps the most significant effect, yet no other researcher has

proposed such an index as a problem parameter. Our empirical results certainly suggest that such an index should be controlled in computational experiments. Further research is apparently warranted to truly characterize this distance effect and to find generation schemes which control it more directly.

CHAPTER VI

OUTLINE OF CONTROLLED COMPUTATIONAL EXPERIMENTS

As reviewed in the literature survey and repeatedly mentioned in this dissertation, a great number of ILP studies attempt to both develop the theory of a proposed procedure and compare the performance of the procedure to that of other algorithms. Comparisons are usually made on the basis of differences in average solution time, average number of simplex iterations, etc. for some small number of test problems. In a strict sense conclusions drawn from such comparisons are valid only for the given test problems. If test problems are randomly generated, much stronger inferences are possible if proper attention is paid to experimental design. However, little work has been reported in the literature on techniques which take full advantage of experimental design for ILP experiments. It is the need for such attention to experimental design that motivated the work of this dissertation, and the outline to follow is the culmination of this research effort.

This outline is addressed to two classes of readers. One class consists of those who intend to do detailed and exact comparison of ILP algorithms. It is suggested that they both follow this outline and read the details at the indicated points in other chapters, because they will probably elect to independently identify significant nuisance parameters, select transformations, etc. The other class consists of those who intend only to roughly compare different algorithms. They are likely to directly use the empirical results desired in the previous

chapters, and can probably obtain sufficient information by reading only this summary.

6.1 General Concept of Controlled Computational Experiments

Controlled experiments are experiments in which nuisance factors not of main concern (but affecting the response) are controlled through experimental design. The advantage of such experiments is improved information about the factors of interest by reducing the effects of nuisance factors. A number of problem characteristics (number of rows, number of columns, etc.) are suspected to greatly affect algorithm performance in ILP experiments. An ideal framework for comparing algorithms is to be able to control the settings of this type of nuisance factor and at least partially eliminate their effects. Such a framework was illustrated in Figure 1, which is reproduced here as Figure 15.

Nuisance Factors / Algorithms	Density (H)				Density (L)			
	m(H)		m(L)		m(H)		m(L)	
	n(H)	n(L)	n(H)	n(L)	n(H)	n(L)	n(H)	n(L)
Algorithm 1								
Algorithm 2								
.								
.								
.								
Algorithm P								

H = High Level and L = Low Level

Figure 15. A Framework for Comparison of ILP Procedures

6.2 Selection of Response Variables

In an empirical study of ILP solution procedures, the first design question to be answered is "What response variables should be used?" A number of response variables have been suggested in this dissertation, all derived from the principle that the total time or effort to obtain an optimal ILP solution in a LP-based algorithm involves an initial LP phase and a series of LP subproblems. Treatment of each subproblem, in turn, involves effort to restart the LP algorithm and post-optimality analysis to generate cuts or calculate bounds. The selection among these response variables is largely dependent upon the degree of interference in comparisons caused by differences in computing machines, LP codes, and strategies for post-optimality analysis. Table 18 is a summary of the results of our discussion. See Sections 4.1.1 and 4.1.2 for detailed discussion.

6.3 Generation of Test Problems

In comparing ILP procedures the researcher's interest is primarily in differences in the efficiency of the "integer aspect" of his or her algorithm. Thus in a sense LP codes, computing machines, and nuisance problem parameters are all nuisances to the experimenter. However, it is economically infeasible for most researchers to run experiments on a variety of computers and LP codes. It is possible, however, that problem nuisance characteristics can be controlled by careful random generation of test problems.

In Chapter III a parametric, random generator for all-integer ILP problems was developed which can control a set of parameters, including most of those the literature of integer programming suggests

Table 18. Selection of Response Variables

	Category 1 (Changes in post-optimality times only)		Category 2 (Changes in N)	
	Same LP	Different LP's	Same LP	Different LP's
Same computer	Total integer time (4-3) or equivalent iterations (4-5)	Total post-optimality time (4-6)	Total integer time (4-3) and LP sub-problems (N) or equivalent iterations (4-5) and LP subproblems (N)	Total integer time (4-3) and LP sub-problems (N)
Different computers	Equivalent iterations (4-5)	Post-optimality time (4-6) adjusted by standard timing factors	Equivalent iterations (4-5) and LP subproblems (N)	Total integer time (4-3) adjusted by standard timing factors and LP sub-problems (N)

relevant to the difficulty of ILP problems. Besides producing feasible, bounded problems with integer constraint coefficients the generator controls to at least some degree the following parameters.

1. Distance index between LP and IP optimality.
2. Degree of dual degeneracy in the LP optimal basis.
3. Degree of primal degeneracy in the LP optimal basis.
4. Density of nonzero entries in the constraint matrix.
5. Determinant of LP optimal basis.
6. Total number of integer variables.
7. Total number of constraints.

As much as possible all other problem characteristics are randomized.

A Fortran program for the generator is listed in Appendix A.

Researchers interested in other problem parameters or in ILP's with special structure may choose to develop their own problem generators. However, the principle of controlling those factors the literature suggests affect problem difficulty and randomizing all other problem elements is appropriate for all random problem generators.

6.4 Selection of Experimental Design

The type of computational experiments proposed in this dissertation are those where several ILP algorithms can be arrayed against classes of test problems characterized by nuisance parameters. See Table 2 for a 2 x 2 illustration. Experiments arranged in this way use problems which cannot be compared across different levels of nuisance parameters. Within a given level of nuisance parameters there will be a variation among randomly generated problems, but this variation can only be understood with respect to the particular level of nuisance parameters. Moreover, if different problems are used in each cell, the variation due to problems will also be incomparable across algorithms. Such an experimental phenomenon is known as nesting.

Within the nesting design format, two factorial design approaches are reasonable. One design blocks the experiment by using the same test problems on algorithms. (See Table 3 for an illustration of this experimental layout.) The second possible design is the completely randomized design where different problems are used in each cell and each replicate (see Table 5 for an illustration of this experimental layout.)

Analysis in Section 4.4 shows that the choice between the blocked

and the randomized design depends on the experimenter's purpose. When his principal objective is the comparison of algorithms, it appears he should elect to block. On the other hand, if the researcher's goal is to measure the effect of nuisance parameters, the analysis suggests the completely randomized design is preferable.

6.5 Selection of Nuisance Parameters to Be Varied

As indicated in the empirical study of Chapter V, nuisance parameter effects may vary from algorithm to algorithm. Thus for those who intend to do exact and detailed comparison, a discussion is presented in Section 6.5.1 of schemes for checking parameter significance. Those who seek only rough answers are referred to the summary of our empirical results summarized in Section 6.5.2.

6.5.1 Empirical Determination of Significant Nuisance Parameters

It is noted in Section 6.4 that the fully randomized factorial design appears preferable when studying the significant nuisance parameters. The empirical study in Chapter V is a large illustration of such a design. However, since the typical experimenter's objective will usually only be the identification of nuisance parameters to control in algorithm experiments, smaller screening experiments would probably be enough for many purposes. For the purpose of ILP experiments, fractional factorial designs (see Box and Hunter (8)) with no blocking on random seeds are suggested for screening experiments. To further simplify the experiment, it appears workable to use time limits or similar censor points to replace censored observations in screening experiments.

6.5.2 Direct Use of Our Results on Nuisance Parameters

Since the two ILP codes selected in our empirical study are representative of the cutting plane (MIF) and branch-and-bound (BB) classes of LP-based ILP algorithms, it appears reasonable to directly use empirical results on the significance of nuisance parameters obtained in Section 5.4. Table 19 shows which parameters these results imply should be varied in ILP experiments for both MIF and BB classes of algorithms. If algorithms being compared are all branch-and-bound type of algorithms, parameters with asterisks under column BB should be varied; similarly, in the case with cutting plane type of algorithms those with asterisks in the MIF column should be varied. If algorithms being compared contain both types, however, all problem parameters should be used.

Table 19. Nuisance Problem Parameters to be Varied

Problem Parameters	BB Type Algorithm	MIF Type Algorithm
1. Distance Index	*	*
2. Dual Degeneracy	*	
3. Primal Degeneracy	*	
4. Density	*	*
5. Determinant		*
6. No. of Variables	*	*
7. No. of Constraints	*	*

6.6 Treatment of Censored Data and ANOVA with Censored Observations Estimated

It is often reported in the literature that particular ILP test problems could not be solved within a specified time limit. Such data are called Type I censored observations in the statistical literature. Sampford and Taylor developed a method for estimation of such censored observations in randomized block experiments and noted that the computational method could be extended to any design for which estimates of location parameters were linear functions of the observations. See Section 4.2.1 for a detailed discussion.

However, in Section 4.2.2 analysis shows that a direct generalization of the Sampford and Taylor procedure to the full factorial case breaks down when all values in a cell are censored. Empirical results in Table 11 show that it is highly likely that all values would be censored at high levels of the nuisance parameters in ILP computational experiments. One way around this difficulty, proposed in Section 4.2.2, is to apply the Sampford and Taylor procedure to one replicate at a time, considering each replicate of the factorial design as a randomized block design. In so doing a replication effect may be caused. Replications essentially become new blocks and the analysis of the experiment must be adjusted accordingly.

The basic tool of all data analysis proposed in this dissertation is the analysis of variance. Little work has been found in the statistical literature on the treatment of censoring in the analysis of variance context. In Section 4.3.2, approximate ANOVA procedures are proposed leading to F-tests derived by analogy. It must be stressed, however, that a value near an operative significance level of the

F-distribution in the approximate test with compared observations should be accepted as significant at that level only with reservations.

6.7 Data Transformation

The technique proposed in this dissertation for data analysis is the analysis of variance (ANOVA). One factor largely affecting the power of the ANOVA is the accuracy of the assumptions in the model underlying the technique. This data analysis method requires independent, equal variance, and normal residual errors. The Sampford and Taylor procedure for treatment of censored data also possesses such requirements.

Verification or satisfaction of these requirements is necessary. However, in the ILP experiments where test problems are randomly generated, the requirement of error variance independence should be automatically met. Data transformation is suggested if one or both of the other two requirements are not satisfied.

Similar to Section 6.5, discussions of the checking of these assumptions are separated on the basis of how exact and detailed comparison of ILP algorithms is to be conducted.

6.7.1 Empirical Determination of Data Transformation

The empirical development of Section 5.3.3 clearly suggests that ANOVA assumptions will not be satisfied without a rather severe data transformation. While a number of methods have been proposed to choose a suitable transformation, it is proposed to use the method by Box and Cox (6). (The development of their procedure is briefly described in Section 4.3.) The family of transformations covered is

$$y^{(\lambda)} = \begin{cases} y^{\lambda} & (\lambda \neq 0) \\ \log y & (\lambda = 0) \end{cases}$$

where y and $y^{(\lambda)}$ are the column vectors of original and transformed observations, and λ is a parameter which characterizes the transformation. Since the maximum log likelihood $L_{max}(\lambda)$ for the observations $y^{(\lambda)}$ can be expressed as a function of λ, the best value of λ may thus be determined by plotting $L_{max}(x)$ against λ for a trial series of values.

A practical problem in using this procedure arises when the Sampford and Taylor method for censored values is also used. A transformation must be applied before censored observations can be estimated, but choice of that transformation depends on the censored values. Empirical results in Chapter V suggest that adequate results will be obtained if the transformation is chosen with censor points (usually time limits) used in place of the censored values.

6.7.2 Direct Use of Our Results on Data Transformations

Since the two ILP codes used in this dissertation are representative of LP-basis ILP procedures, our empirical results on data transformation may be used for rough comparison. Table 20 shows the λ values for the transformations used in the empirical study of Chapter V. The

Table 20. Transformation Used in Empirical Study

	For Sampford and Taylor Procedure	For Application of Analysis of Variance
Cutting Plane	-.1	5.2
Branch and Bound	-.05	2.2

results are applied by first applying the transformations in the left column to observed data; then estimating (transformed) values for the censored observations; then applying the transformations in the right column to prepare all data for the analysis of variance.

CHAPTER VII

CONCLUSIONS AND RECOMMENDATIONS

This dissertation is the culmination of a research effort which attempted to develop a general approach to conducting controlled computational experiments on integer linear programming (ILP) procedures. Chapters III and IV presented and discussed analytically some important issues related to the development of such an approach. In Chapter V an empirical study was performed to gain insights into the analysis of Chapters III and IV. A brief outline of the experimental approach implied by this research was reported in Chapter VI. The main results of the complete research effort can be summarized as follows.

1. Development of Parametric Random Problem Generator for ILP Problems. No systematic scheme for parametric random problem generation is accepted in the ILP literature. The generator developed in this research provides such scheme by controlling the parameters which ILP researchers have indicated most affect computational efficiency of ILP procedures. Besides generating feasible and bounded problems with integer coefficients, the generator at least partially controls the determinant of the optimal LP basis, the number of variables, the number of constraints, the density of nonzero entries in the constraint matrix, the degree of primal and dual degeneracy in the LP solution, and the distance between the optimal LP and ILP solutions. Since it does control these key problem characteristics, the generator can serve as a source of test problems for both algorithm comparisons and tests of

parameter effects in a wide range of integer programming research.

2. Empirical Study of What Parameters Make ILP Problems Hard to Solve. A number of studies have conjectured and advocated particular problem parameters as the reason that ILP problems are hard to solve. This dissertation research conducted an extensive empirical study which employed the analysis of variance to confirm or refute the various views. Interestingly, the statistical results from this study seem to confirm many of the observations about ILP complexity made individually by other research. Jeroslow (30), for example, advocated the determinant of the LP optimal basis as the most important parameter. While his view was not completely confirmed, our results did indicate that the determinant effect is highly significant for the cutting plane algorithm. Similarly, the recent observations by Geoffrion and others (18) that more constraints may make ILP problems easier to solve was confirmed by the results for both branch-and-bound and cutting plane algorithms.

3. Development of a Design for Controlled Computational Experiments on ILP Solution Procedures. Although testing and comparison of algorithms is an integral part of algorithm development, little work has been reported in the literature on techniques for the conduct of this vital exercise. When test problems are randomly generated, the field of statistical experimental design provides a base on which to develop such techniques. A key part of this research has been the specialization of the methods of statistical analyses and experimental design to deal with some of the peculiar problems of computational experiments in the algorithms versus nuisance parameter framework. It is hoped that the resulting approaches provide both a more powerful

means of evaluating and comparing ILP procedures than is presently available and the beginning of a whole body of research on the design of computational experiments in mathematical programming.

At the completion of this research, a wide range of possibilities for extensions are apparent. Such possibilities are best discussed in the following three categories.

1. Development of Similar Approaches to Other Mathematical Programming Experiments. There is an amerging evidence that researchers in the field of mathematical programming will have to use randomly generated test problems as they undertake large scale experiments with proposed algorithms. Thus statistical techniques may provide the future basis of experimental design in many areas of mathematical programming. It would therefore appear fruitful to pursue an analogy to the research in this dissertation in say the context of nonlinear programming or large-scale linear programming. The only elements of the development likely to vary widely from our ILP results are the nuisance parameters used to control problem difficulty and the corresponding random problem generator.

2. Development and Further Studies with the ILP Problem Generator. Our empirical research with the present version of the ILP problem generator suggests some areas for further development and experimentation. In particular, the empirical results of Chapter V strongly indicate that the index of distance between the LP and ILP optimal solution used in the current generator played a significant role in computational complexity of the resulting test problems. It is thus suggested that further research is warranted to truly

characterize this distance effect and to find generation schemes which control it more directly. Another possibility in this category is to investigate the effect of the subgroup structure in the optimal LP basis. Gomory and others have noted that this subgroup structure may be a significant factor in the underlying character of ILP problems. It is conjectured that this factor could be incorporated without a complete restructuring of the generator of Chapter III because the subgroup structure is related to the factorization of the determinant along the diagonal matrix of the Smith normal form.

3. Possibilities Related to ILP Experimental Design. In this research the selection of response variables could not be completely investigated in empirical studies because economic and time limitations prohibited a study involving a variety of ILP algorithms, LP algorithms, and computers. Nevertheless, lack of adequate measures of ILP algorithm efficiency is a severe limitation on the development and comparison of ILP procedures in the integer programming literature. Larger empirical studies would thus seem warranted.

A second area of experimental design needing further attention is the handling of censored data. The censoring problem is not as important in most statistical experiments as it is in computational experiments, and thus the available literature is quite limited. The Sampford and Taylor procedure used in this dissertation appears to give adequate results, but it is complex, and practically nothing is known about the properties of the approximate F-tests to which the procedure leads. More study of the statistics of censored data in the analysis of variance context is needed.

It would also be interesting to study whether much simpler procedures such as using the censor points in lieu of the censored observations would give satisfactory results. Since the censor points are largely set by the researcher in choosing his time limit, this issue should be addressed from the standpoint of additional information gained for a given investment of time and money. Longer time limits would reduce censoring, but at considerable computer cost. This dissertation used four minutes at the CPU time limit and ran two replicates; however, one replicate might be sufficient if six minutes were used.

Finally, it would be useful to obtain empirical verification of the tentative conclusion in Section 4.4 that it is more efficient to block on random number seeds when the purpose of an experiment is comparison of algorithms. The purpose of the large empirical experiments in Chapter V was investigation of the effects of nuisance parameters, and thus blocking was not used. However, parallel experiments could be run (with random seeds blocked across algorithms) that would permit the direct estimation of the problem and problem-algorithm variances which figure importantly in the analysis of Section 4.4.

APPENDIX A

THE PROBLEM GENERATION CODE

APPENDIX A

THE PROBLEM GENERATION CODE

A computer code of the generating procedure for test problems was written in Fortran and was implemented for use on the Univac 1108 at the Georgia Institute of Technology and used in all empirical work of this dissertation. A listing of the computer code follows this introduction.

The code can generate ILP test problems having up to 40 constraints and 80 integer variables. Larger problem sizes can be generated by changing the dimension of the program.

The procedure requires the use of a random number generator to obtain the elements of the vectors and matrices. The pseudo-random number generator used is known as the multiplicative congruential method. The general procedure is described as follows:

Let

$$Z_n = a \cdot Z_{n-1} \pmod{m}$$

where a is a constant multiplier

m is the modulus, and

Z_o is the initial random seed.

The sequence, Z_n, approximates a sequence of integers uniformly distributed between zero and the modulus. Then numbers on the unit interval 0,1 are formed by

$$U_i = Z_i/m$$

On the Univac the convenient number 2^{35} was used with corresponding $a = 3125$.

In order to eliminate the possibility of excess round-off errors in the solution of the ILP, a routine is included in the code so that the values of the nonzero entries in the basis matrix B are small, and those of the entries in N are no larger than the largest absolute value of the entries in B. For similar reasons, the method of generating c_B and c_N in steps 11 and 12 of Section 3.2 was slightly modified. Components in $(c_N - c_B B^{-1} N)$ which were supposed to actually equal 0 (because of dual degeneracy) were allowed to take on small positive fractional values. Thus the code actually generates only "near" dual degeneracy but the magnitude of components in c_B can be reduced.

The near degeneracy concept also prevented an additional problem which arises when true dual degeneracy is present. When these are alternative optional bases for the LP relaxation of ILP, there is no guarantee the one which was carefully structured by the generator will actually be chosen by the LP algorithm. However, if the problem is only close to dual degeneracy the optimal LP basis is unique.

```
      C
      C  THIS PROGRAM IS TO RANDOMLY GENERATE INTEGER PROGRAMMING PROBLEMS
      C
      C     M= NO OF CONTRAINTS
      C     N= NO OF VARIABLES
      C     CSTMAX,CSTMIN = RANGE OF COST
      C     IY = RANDOM SEED
      C     IPMATX=INITIAL DIAGONAL ELEMENTS
      C     IPMILP=DISTANCE INDEX
      C     INT2=DEGREE OF PRIMAL DEGENERACY
      C     INT3=DEGREE OF DUAL DEGENERACY
      C
      C
            REAL IBMATX,IBMAT1,MULTI,IDUM,JBMATX,JBMAT2
            COMMON MULTI(40,80) ,M,N,IY,IY1,IY2,IY3,IY4,IY5,INTX(80)
            INTEGER CSTMAX,CSTMIN
            DIMENSION  COST(80)  ,RRIGHT(40),IPTMAX(40),
           1 JBMAT1(40,40), JBMAT2(40,40), JBMATX(40,80),
           2 XN(40),IND(80),INDX(40),  IBMATX(40,40) ,IBMAT1(40,40)
           3,RIGHT(40),SUM5(40,40) ,RIGHT1(40)
           4,IPTMA1(40) ,IDUM(40,40) ,BBMATX(40,80)
            REWIND 12
            READ(5,10) IY,IY1,IY2,IY3,IY4,IY5
            READ(5,20) M,N,IPMILP
            READ(5,20) NUM,INT1
            READ(5,20) INT2,INT3,IDUAL
            READ(5,20)  CSTMAX,CSTMIN,IPTM1
            ISTART =1
         39 IEND = ISTART + 5
            IF (IEND .GT. M) IEND = M
            READ(5,20) (IPTMAX(I) ,I=ISTART,IEND)
            IF (IEND .EQ. M) GO TO 41
            ISTART = IEND +1
            GO TO 39
         41 CONTINUE
      C
      C
            DO 60 J=1,M
            DO 50 K=1,M
            IF (J.NE.K) GO TO 50
            IBMATX(J,J)= 1.0/FLOAT(IPTMAX(J))
            JBMATX(J,J) = IPTMAX(J)
            JBMAT2(J,J) =IPTMAX(J)
            IBMAT1(J,J) =FLOAT(IPTMAX(M))/FLOAT(IPTMAX(J))
            GO TO 60
         50 CONTINUE
            JBMATX(J,K) = 0
            JBMAT2(J,K) = 0
            IBMATX(J,K) =0.0
            IBMAT1(J,K) = 0
         60 CONTINUE
      C
            IDETER= 1
            DO 70 I= 1,M
         70 IDETER =IDETER *IPTMAX(I)
      C
```

```
      C     GENERATE PRIMAL DEGENERACY ROWS
      C
            CALL IND1(IND,1,M,IY)
            INTT = 0
              DO 80 I=1,M
            IF (INTT .GE. INT2) GO TO 90
            II =IND(I)
            INDX(II) =1
            INTT =INTT +1
         80 CONTINUE
         90 CONTINUE
      C
      C     GENERATE OPTIMAL SOLUTION AND RIGHT HAND SIDE
      C
            DO 105 I=1,M
        110 IF (IPTMAX(I) .LE. 100) GO TO 125
            IP =IPTMAX(I)
        120 IP1 =FLOAT(IP) /FLOAT(IPTM1)
            IPTMA1(I) = IP1
            IF (IP1 .LE. 100) GO TO 130
            IP =IP1
            GO TO 120
        125 CONTINUE
            IPTMA1(I) =IPTMAX(I)
        130 CONTINUE
            IFAC =IPTMA1(I) *UNIF(IY) +1 +IPTMA1(I)
            RRIGHT(I) =1 *IFAC
            XN(I) =FLOAT(RRIGHT(I)) /FLOAT(IPTMA1(I))
            JXN=XN(I)
            XNF=XN(I)-JXN
            IF (XNF .NE. 0.0 .OR. IPTMA1(I) .EQ. 1) GO TO 140
            RRIGHT(I) = RRIGHT(I) - 1
            XN(I) = FLOAT(RRIGHT(I))/FLOAT(IPTMA1(I))
        140 CONTINUE
            IF (INDX(I) .NE. 1) GO TO 105
            XN(I) = 0.0
            RRIGHT(I) = 0
        105 CONTINUE
      C
            WRITE(6,150)
        150 FORMAT(  '   THE GENERATED LP OPTIMUM SOLUTION')
            WRITE(6,160) (XN(I),I=1,M)
        160 FORMAT(1H,18F7.4)
            M1 =M-1
      C
            DO 180 I=1,M
            DO 170 J=1,M
        170 IDUM(I,J) = 0
            IDUM(I,I) =1
        180 CONTINUE
            ITERA = 0
      C
      C
            INTT = 0
            DO 205 L=1,20
            DO 205 I=1,M
            JSIGN =2* UNIF(IY1) +1
```

```
114             GO TO (220, 230) ,JSIGN
115         220 ISIGN = 1
116             GO TO 240
117         230 ISIGN = -1
118         240 CONTINUE
119             IROW2 =I
120         250 IROW1 =M*UNIF(IY1) +1
121             ITERA =ITERA + 1
122             IF (IROW1 .EQ. IROW2) GO TO 250
123             RATIO =FLOAT(IPTMAX(IROW2))/ FLOAT(IPTMAX(IROW1))
124             IF (IROW1 .LT. IROW2) GO TO 260
125             IF (M.GT.2) GO TO 270
126             IF(RATIO.LE. .05) GO TO 205
127             GO TO 280
128         270 IF (RATIO .LE. .1) GO TO 250
129         280 CONTINUE
130             IFA1 =1 +2 *UNIF(IY1)
131             IRATIO =FLOAT(IPTMA1(IROW1)) /FLOAT(IPTMA1(IROW2))
132             IJK =IRATIO *IFA1
133             GO TO 320
134         260 CONTINUE
135             IF (M.GT. 2) GO TO 290
136             IF (RATIO.GE.1000) GO TO 205
137             GO TO 300
138         290 IF (RATIO .GE. 10) GO TO 250
139         300 CONTINUE
140             IRATIO =FLOAT(IPTMA1(IROW2)) /FLOAT(IPTMA1(IROW1))
141             IJK =2*UNIF(IY1) +1
142             IFA1 = IJK *IRATIO
143         320 CONTINUE
144             CALL MULTI2(MULTI,M,IJK,ISIGN,IROW1,IROW2)
145             CALL OPMATX(M,IDUM,MULTI,M,M)
146             CALL MULTI2(MULTI,M,IFA1,ISIGN,IROW1,IROW2)
147             CALL OPMATX(M,JBMATX,MULTI,M,M)
148             CALL OPMATX(M,JBMAT2,MULTI,M,M)
149             ISIGN =-ISIGN
150             CALL MULTI2(MULTI,M,IFA1,ISIGN,IROW1,IROW2)
151             CALL PREMUL(IBMATX,MULTI,M,M)
152             CALL PREMUL(IBMAT1,MULTI,M,M)
153             INTT =INTT + 1
154             IF (INTT .GE. INT1 ) GO TO 207
155         205 CONTINUE
156       C
157         207 CONTINUE
158             CALL PREMUL(RRIGHT,IDUM,M,1)
159             DO 350 I=1,M
160             IRATIO =FLOAT(IPTMAX(I)) /FLOAT(IPTMA1(I))
161             IF (IRATIO .LE. 1) GO TO 350
162             RRI  = IRATIO * RRIGHT(I)
163             RRIGHT(I) = RRI
164         350 CONTINUE
165       C
166             NITER= M*M
167             MMMM = M*2
168       C
169       C    RANDOMLY MULTIPLY BASIC MATRICES
170       C
```

```
171             ITERA=0
172         400 CONTINUE
173             CALL LARGE(JBMAT2, I1 ,J1,LARG1,M)
174             CALL LARGE(JBMATX, I2, J2, LARG,M)
175             JABS=IABS(LARG )
176             IF(ITERA .GE. MMM) GO TO 510
177             IF(IABS(LARG1) .LE. 4) GO TO 510
178             IF (JJ22 .EQ. J1) GO TO 480
179             DO 470 I=1,M
180             IF (JBMATX(I,J1) .EQ. 0.OR. I.EQ.I1 ) GO TO 470
181             IF (JBMATX(I1,J1) .GT. 0.AND.JBMATX(I,J1) .GT.0) GO TO 420
182             IF (JBMATX(I1,J1) .LT. 0.AND.JBMATX(I,J1) .LT. 0) GO TO 420
183             DO 410 J=1,M
184             JBMAT1(I1,J) =JBMATX(I1,J) +JBMATX(I,J)
185         410 CONTINUE
186             ISIGN =1
187             GO TO 440
188         420 CONTINUE
189             DO 430 J=1,M
190             JBMAT1(I1,J) =JBMATX(I1,J) -JBMATX(I,J)
191         430 CONTINUE
192             ISIGN =-1
193         440 CONTINUE
194             MMM=0
195             DO 450 J=1,M
196             IF(IABS(JBMAT1(I1,J)) .LE. IABS(MMM)) GO TO 450
197             MMM =JBMAT1(I1,J)
198         450 CONTINUE
199             IF(IABS(MMM) .GT. IABS(LARG)) GO TO 470
200             DO 460 J=1,M
201             JBMATX(I1,J) =JBMAT1(I1,J)
202         460 CONTINUE
203             IROW1=I1
204             IROW2=I
205             GO TO 490
206         470 CONTINUE
207         480 CONTINUE
208             JJ22 = J1
209             JBMAT2(I1,J1) =0
210             ITERA =ITERA +1
211             GO TO 400
212         490 CONTINUE
213             DO 500 I=1,M
214             DO 500 J=1,M
215             JBMAT2(I,J) =JBMATX(I,J)
216         500 CONTINUE
217             CALL MULTI2(MULTI,M,1,ISIGN,IROW1,IROW2)
218             CALL PREMUL(RRIGHT,MULTI,M,1)
219             ISIGN=-ISIGN
220             CALL MULTI2(MULTI,M,1,ISIGN,IROW1,IROW2)
221             CALL OPMATX(M,JBMATX,MULTI,M,M)
222             CALL OPMATX(M,JBMAT1,MULTI,M,M)
223             ITERA =ITERA +1
224             GO TO 400
225         510 CONTINUE
226         C
227         C     ENSURE EACH COLUMN HAVING MORE THAN ONE ELEMENT
```

```
228       C
229             DO 600 I=1,M
230             IIND = 0
231             IIND1 = 0
232             DO 605 J=1,M
233             IF (JBMATX(I,J) .EQ. 0) GO TO 605
234             IIND = IIND +1
235           605 CONTINUE
236             IF (IIND .GT. 1) GO TO 600
237           610 JSIGN = 1
238             GO TO ( 620,630) ,JSIGN
239           620 ISIGN= 1
240             GO TO 640
241           630 ISIGN =-1
242           640 CONTINUE
243           650 IROW2 = M*UNIF(I/2) + 1
244             IF (IROW2 .EQ. I) GO TO 650
245             CALL MULTI2(MULTI,M,1,ISIGN,I,IROW2)
246             CALL PREMUL(JBMATX,MULTI,M,M)
247             CALL PREMUL(RRIGHT,MULTI,M,1)
248             ISIGN =-ISIGN
249             CALL MULTI2(MULTI,M,1,ISIGN,I,IROW2)
250             CALL OPMATX(M,IBMATX,MULTI,M,M)
251             CALL OPMATX(M,IBMAT1,MULTI,M,M)
252             IIND1 = IIND1 + 1
253             IF (IIND1 .LT. 2) GO TO 610
254           600 CONTINUE
255       C
256             MAXABS =0
257             DO 700 I=1,M
258             DO 700 J=1,M
259             IF(IABS(JBMATX(I,J)).GT.MAXABS) MAXABS=IABS(JBMATX(I,J))
260           700 CONTINUE
261       C
262             NMM = N -M
263             DO 800 J=1,M
264             DO 800 K=1,NMM
265             K1= M +K
266         800 JBMATX(J,K1) =0
267       C
268       C     CALCULATE NO OF ELEMENTS NEEDED IN NONBASIC MATRIX
269       C
270             JFIX = UNIF(IY5) *NMM + M +1
271             CALL DNSITY(JBMATX,M,M,NNZERO,DNZERO)
272             DO 900  J = 1,NMM
273             NMM1 = UNIF(IY3) * M + 1
274             J1=J+M
275             IN = -.5*MAXABS
276             INTNUM = UNIF(IY3) * IN + MAXABS* .3 +0
277             IF (INTNUM .EQ. 0) INTNUM =1
278             JBMATX(NMM1,J1) = INTNUM
279           900 CONTINUE
280             NUMZRO =NUM -NNZERO-(N-M) - (M -1)
281       C
282       C     GENERATE ELEMENTS IN NONBASIC MATRIX
283       C
284             NUMZ = 0
```

```
285             DO 1000 IDUMMY = 1,1000
286             NMM1 = UNIF (IY) *M + 1
287             J1 =UNIF(IY) * NMM +1 + M
288             IF (JBMATX(NMM1 ,J1) .NE. 0 .OR. J1.EQ. JFIX) GO TO 1000
289             IN = -.5 *MAXABS
290             INTNUM = UNIF(IY) *IN + MAXABS * .3
291             IF (INTNUM .EQ. 0) INTNUM = 1
292             JBMATX(NMM1,J1) = INTNUM
293             NUMZ = NUMZ + 1
294             IF (NUMZ .GE. NUMZRO ) GO TO 1010
295          1000 CONTINUE
296          1010 CONTINUE
297             M1=M +1
298             DO 1020 J=M1,NMM
299             INTX(J) = IPMILP
300          1020 CONTINUE
301             DO 1030 J=1,M
302             INTX(J) =XN(J)
303          1030 CONTINUE
304             DO 1050 I=1,M
305             ISUM= 0
306             M1 =M+1
307             DO 1040 J= 1,N
308             ISUM = ISUM + JBMATX(I,J) *INTX(J)
309          1040 CONTINUE
310             JBMATX(I,JFIX) = RRIGHT(I) -ISUM
311          1050 CONTINUE
312          C
313          C     GENERATE COST COEFFICIENTS FOR BASIC COLUMNS
314          C
315             DO 1060 J=1,M
316             ICST =UNIF(IY5) *(CSTMAX-CSTMIN)
317             COST (J) =CSTMIN +FLOAT(ICST)
318          1060 CONTINUE
319          C
320          C     GENERATE COST COEFFICIENTS FOR NONBASIC COLUMNS
321          C
322             DO 1075 I=1,M
323             DO 1075 K=1,NMM
324             SUMM =0
325             SUM1=0
326             K1 =K+M
327             DO 1070 J=1,M
328             SUMM=  SUMM+ IBMAT1(I,J)  * JBMATX(J,K1)
329             SUM1 = SUM1 +IBMATX(I,J) *RRIGHT(J)
330          1070 CONTINUE
331             SUM5(I,K1) =SUMM
332             RIGHT(I) = SUM1
333          1075 CONTINUE
334          C
335          C     GENERATE DUAL DEGENERACY COLUMNS
336          C
337             M1 = M+1
338             CALL IND1(IND,1,NMM,IY4)
339             INTT = 0
340             DO 1080 I=1,NMM
341             IF (INTT .GE. INT3) GO TO 1090
```

```
342             II =IND(I) + M
343             INDX(II) =1
344             INTT =INTT+1
345          1080 CONTINUE
346          1090 CONTINUE
347           C
348           C
349                DO 1100 I2 = 1,NMM
350                I=I2+ M
351                SUM2 = 0.0
352                DO 1110 J = 1,M
353                SUM2 = SUM2 + COST(J) * SUMS(J,I)
354          1110 CONTINUE
355                ISSUM =10*UNIF(IY4)
356                IF (INDX(I) .NE. 1) GO TO 1120
357                IF (IDUAL .EQ. 0) DUAL= .00008
358                IF (IDUAL .EQ. 1) DUAL = .00016
359                IF (IDUAL .EQ. 2) DUAL= .00100
360          1115 COST(I) = SUM2 -DUAL
361                GO TO 1130
362          1120 CONTINUE
363                COST(I) =SUM2 -FLOAT(ISSUM) -10.
364          1130 CONTINUE
365                DO 1140  I= 1,M
366                ICST=COST(I) *IPTMAX(M)
367          1140 COST(I) = ICST
368           C
369           C
370                DO 1160 I=1,M
371                DO 1150 J=1,N
372          1150 BBMATX(I,J) =UBMATX(I,J)
373          1160 RIGHT1(I) =RRIGHT(I)
374           C
375           C     WRITE INTO UNIT 12
376           C
377                IZERO =0
378                ISTART = 1
379          2000 IEND = ISTART + 3
380                IF (IEND .GT. N) IEND = N
381                WRITE(12,100)(J,IZERO , COST(J), J= ISTART,IEND)
382                IF( IEND .EQ. N) GO TO 2010
383                ISTART = IEND + 1
384                GO TO 2000
385          2010 CONTINUE
386                WRITE(6,2020) (COST(J),J=1,N)
387          2020 FORMAT(1H0,'**' ,15F8.3)
388                WRITE(12,101)
389                ISTART = 1
390          2030 IEND = ISTART + 3
391                IF (IEND .GT. M) IEND = M
392                WRITE(12,100)(J,IZERO ,RIGHT1(J) ,J=ISTART,IEND)
393                IF( IEND .EQ. M) GO TO 2040
394                ISTART = IEND + 1
395                GO TO 2030
396          2040 CONTINUE
397                WRITE(12,101)
398                DO 2060 I= 1,M
```

```
399           ISTART = 1
400        2050 IEND = ISTART +2
401           IF (IEND .GT. N) IEND = N
402           EI = I
403           WRITE(12,102)EI,(J,IZERO,BBMATX(I,J), J= ISTART,IEND)
404           IF( IEND .EQ. N) GO TO 2060
405           ISTART = IEND + 1
406           GO TO 2050
407        2060 CONTINUE
408           WRITE(12,101)
409           WRITE(6,101)
410         C
411         C
412           M1= M+1
413           I=1
414        2070 IF(I.EQ. M1) GO TO 2080
415           RR=RRIGHT(I)
416           WRITE(6,2090)  (JBMATX(I,J) ,J=1,N), RR
417           I=I+1
418           GO TO 2070
419        2080 CONTINUE
420        2090 FORMAT(1H,'**',8F8.1)
421         C
422           CALL DNSITY(JBMATX,M+1,NNZERO,DNZERO)
423           WRITE(6,32) DNZERO
424             12 FORMAT(I10)
425             16 FORMAT(6(I11,1X))
426             20 FORMAT(10I6)
427             32 FORMAT(1H,  'DENSITY OF PROBLEM =',F7.3)
428            100 FORMAT(4(I3,I1,F10.3))
429            113 FORMAT(3(I3,I1,I6))
430            101 FORMAT('9999999999.999')
431            102 FORMAT(4X,F10.3, 3(I3,I1,F10.3))
432           END FILE 12
433           STOP
434           END
```

```
  1           SUBROUTINE DNSITY( AMATX, NROW1,NROW2, NNZERO,DNZERO )
  2           DIMENSION AMATX(40,80)
  3           NNZERO = 0
  4           DO 15 I= 1,NROW1
  5           DO 15 J= 1,NROW2
  6           IF (AMATX(I,J) .EQ. 0) GO TO 15
  7           NNZERO = NNZERO + 1
  8          15 CONTINUE
  9           DNZERO =FLOAT(NNZERO)/FLOAT(NROW1*NROW2)
 10           RETURN
 11           END
```

```
      SUBROUTINE IND1(IND,LL,MM,IY)
      DIMENSION IND(80) ,IWK(80)
      LL1 =LL-1
      DO 10 I=LL,MM
   10 IWK(I) = I
      K= MM+1
   20 K=K-1
      IF (K.EQ.LL1) GO TO 35
      IR= UNIF(IY) *K +1
      IND(K) =IWK(IR)
      IF(K.EQ. IR) GO TO 20
      IMAX = K-1
      DO 30 I= IR,IMAX
   30 IWK(I) =IWK(I +1)
      GO TO 20
   35 RETURN
      END
```

```
      SUBROUTINE LARGE(AMATX ,INDEX1,INDEX2,LARG,M)
      DIMENSION AMATX(40,80)
      LARG =0
      DO 1 I=1,M
      DO  1 J=1,M
      IN1 =M-I +1
      JN1 =M-J +1
      IF (ABS(AMATX(IN1,JN1)) .LE. ABS(LARG )) GO TO 1
      LARG =AMATX(IN1,JN1)
      INDEX1 =IN1
      INDEX2 =JN1
    1 CONTINUE
    2 RETURN
      END
```

```
      SUBROUTINE MULTI2(MULTI , NROW  ,IFACTR ,IPOS,IROW1,IROW2)
      REAL MULTI
      DIMENSION MULTI(40,80)
      DO 15 I= 1,NROW
      DO 15 J= 1,NROW
   15 MULTI(I,J) = 0.0
      DO 16 I= 1,NROW
   16 MULTI(I,I) = 1.0
      MULTI( IROW1, IROW2 ) = IFACTR *IPOS
      RETURN
      END
```

```
      SUBROUTINE PREMUL(OPTMAX,MULTI,M,K1)
      REAL MULTI,IUM
      DIMENSION OPTMAX(40,40) , MULTI(40,80) ,IUM(40,80)
      DO 210 I=1,M
      DO 210 K=1,K1
      DO 210 J=1,M
      IUM(I,K) = MULTI(I,J) * OPTMAX(J,K)+IUM(I,K)
  210 CONTINUE
      DO 220 I=1,M
      DO 220 K=1,K1
      OPTMAX(I,K)  = IUM(I,K)
      IUM(I,K) = 0.0
  220 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE OPMAT2(K2,OPTMAX,MULTI,M,J1)
      REAL MULTI
      DIMENSION OPTMAX(40,80) , MULTI(40,80) ,SUMM1(40,80)
      DO 310 I=1,M
      DO 310 K=1,K2
      DO 310 J=1,J1
      SUMM1(I,K) = OPTMAX(I,J) * MULTI(J,K) + SUMM1(I,K)
  310 CONTINUE
      DO 320 I=1,M
      DO 320 K=1,K2
      OPTMAX(I,K)  = SUMM1(I,K)
      SUMM1(I,K) = 0.0
  320 CONTINUE
      RETURN
      END
```

```
      FUNCTION UNIF(IY)
      IY=IY*3125
      IF(IY)115,116,116
  115 IY=IY+1+34359738367
  116 YFL=IY
      UNIF=YFL*2.**(-35)
      RETURN
      END
```

APPENDIX B

REPLICATE I OF DATA COLLECTED

LEGEND FOR APPENDICES B AND C

1. A: The number of constraints
2. B: The number of integer variables
3. C: The determinant of the LP optimal basis
4. D: The density of nonzero entries in the constraint matrix
5. E: The degree of primal degeneracy in the optimal solution of the LP relaxation
6. F: The degree of dual degeneracy in the optimal solution of the LP relaxation
7. G: The distance index between the optimal solutions to the LP relaxation and ILP
8. H: The number of iterations for solving the LP relaxation
9. I: Time for solving the LP relaxation
10. J: Total number of simplex iterations from the optimal solution of the LP relaxation to IP optimality
11. K: Total time from the optimal solution of the LP relaxation to IP optimality
12. L: The number of LP subproblems

REPLICATE I OF DATA COLLECTED

Problem Characteristics								Branch-Bound					Cutting Plane				
								First LP		From LP to IP			First LP			From LP to IP	
Problem No.	A	B	C	D	E	F	G	H	I	J	K	L	H	I	J	K	L
1	-	-	-	-	-	-	-	17	.142	70	.885	14	12	.107	1	.345	1
2	+	-	-	-	-	-	-	23	.258	62	.913	7	20	.169	1	.516	1
3	-	+	-	-	-	-	-	12	.086	15	.352	4	16	.151	1	.431	1
4	+	+	-	-	-	-	-	31	.368	5	.253	2	30	.375	2	.519	1
5	-	-	+	-	-	-	-	13	.104	1	.107	1	12	.089	1	.342	1
6	+	-	+	-	-	-	-	23	.216	1	.169	1	21	.198	4	.530	1
7	-	+	+	-	-	-	-	15	.142	>23072	>240.000	>3077	13	.125	2	.516	2
8	+	+	+	-	-	-	-	22	.267	147	2.380	16	23	.247	6	.620	1
9	-	-	-	+	-	-	-	18	.168	24	.387	4	21	.184	20	.867	2
10	+	-	-	+	-	-	-	34	.464	941	13.550	66	28	.355	6	.680	1
11	-	+	-	+	-	-	-	23	.317	7	.335	3	20	.318	10	.984	2
12	+	+	-	+	-	-	-	37	.651	1048	22.980	87	32	.563	15	1.249	1
13	-	-	+	+	-	-	-	13	.100	140	1.591	25	15	.134	2	.445	1
14	+	-	+	+	-	-	-	29	.297	10	.631	4	23	.300	3	.580	1
15	-	+	+	+	-	-	-	17	.201	5	.289	3	16	.176	1	.479	1
16	+	+	+	+	-	-	-	45	.762	3300	70.197	236	36	.596	10	1.068	1
17	-	-	-	-	+	-	-	11	.083	2	.112	1	9	.082	2	.373	1
18	+	-	-	-	+	-	-	17	.165	2	.148	1	19	.175	4	.490	1
19	-	+	-	-	+	-	-	10	.089	58	.801	11	11	.075	4	.455	1
20	+	+	-	-	+	-	-	23	.273	29	.650	4	24	.271	3	.543	1
21	-	-	+	-	+	-	-	8	.052	3	.129	2	13	.095	3	.376	1
22	+	-	+	-	+	-	-	19	.117	3	.137	1	15	.104	2	.452	1
23	-	+	+	-	+	-	-	16	.127	19	.387	5	12	.106	4	.491	1
24	+	+	+	-	+	-	-	20	.168	6	.271	3	24	.294	2	.576	1
25	-	-	-	+	+	-	-	19	.130	7	.190	2	13	.134	15	.653	1
26	+	-	-	+	+	-	-	22	.297	1	.184	1	37	.481	2	.578	1

Problem Characteristics								Branch-Bound					Cutting Plane				
Problem No.	A	B	C	D	E	F	G	First LP		From LP to IP			First LP		From LP to IP		
								H	I	J	K	L	H	I	J	K	L
27	-	+	-	+	+	-	-	15	.206	545	8.259	74	30	.375	34	2.438	6
28	+	+	-	+	+	-	-	29	.531	2	.211	1	40	.614	14	1.228	2
29	-	-	+	+	+	-	-	13	.103	30	.596	9	19	.209	11	.857	3
30	+	-	+	+	+	-	-	23	.244	1	.132	1	38	.407	23	1.552	3
31	-	+	+	+	+	-	-	14	.147	47	.794	5	16	.215	4	.589	1
32	+	+	+	+	+	-	-	32	.444	33	.683	2	32	.564	14	1.540	2
33	-	-	-	-	-	+	-	11	.086	1716	18.575	401	13	.100	3	.383	1
34	+	-	-	-	-	+	-	24	.221	2	.181	2	23	.225	1	.416	1
35	-	+	-	-	-	+	-	15	.099	118	1.327	22	12	.130	8	.752	3
36	+	+	-	-	-	+	-	29	.423	529	9.709	63	24	.282	2	.543	1
37	-	-	+	-	-	+	-	10	.086	2655	27.989	592	12	.109	4	.439	1
38	+	-	+	-	-	+	-	19	.173	3	.220	1	19	.198	3	.564	1
39	-	+	+	-	-	+	-	8	.090	2	.187	2	16	.155	9	.662	2
40	+	+	+	-	-	+	-	23	.314	49	1.235	7	29	.415	1	.589	1
41	-	-	-	+	-	+	-	25	.255	2673	34.300	442	15	.107	1	.387	1
42	+	-	-	+	-	+	-	32	.502	71	1.789	9	42	.650	3	.627	1
43	-	+	-	+	-	+	-	25	.279	2517	32.668	315	31	.414	86	4.642	13
44	+	+	-	+	-	+	-	47	.773	136	2.877	10	35	.683	47	4.343	6
45	-	-	+	+	-	+	-	22	.228	17249	228.051	3482	12	.121	24	1.208	4
46	+	-	+	+	-	+	-	29	.417	220	4.710	25	25	.508	19	1.703	2
47	-	+	+	+	-	+	-	17	.184	>18250	>240.000	>1940	24	.302	1	.514	1
48	+	+	+	+	-	+	-	35	.622	>11198	>240.000	>1096	34	.569	32	2.321	2
49	-	-	-	-	+	+	-	13	.081	1	.101	1	10	.099	5	.469	1
50	+	-	-	-	+	+	-	15	.120	3	.165	2	19	.173	3	.427	1
51	-	+	-	-	+	+	-	13	.990	18	.285	3	15	.108	2	.422	1
52	+	+	-	-	+	+	-	23	.274	504	7.333	51	21	.246	2	.612	1
53	-	-	+	-	+	+	-	13	.104	179	1.852	28	14	.123	1	.473	1
54	+	-	+	-	+	+	-	23	.226	1	.130	1	22	.228	4	.496	1
55	-	+	+	-	+	+	-	16	.159	41	.576	8	21	.193	6	.651	2

Problem Characteristics								Branch-Bound					Cutting Plane				
Problem No.	A	B	C	D	E	F	G	First LP		From LP to IP			First LP		From LP to IP		
								H	I	J	K	L	H	I	J	K	L
56	+	+	+	-	+	+	-	19	.213	1	.151	1	23	.257	8	.690	1
57	-	-	-	+	+	+	-	12	.106	247	3.351	33	18	.168	1	.417	1
58	+	-	-	+	+	+	-	32	.314	56	1.048	7	28	.395	5	.695	1
59	-	+	-	+	+	+	-	27	.297	1	.125	1	19	.157	1	.572	1
60	+	+	-	+	+	+	-	27	.384	2	.234	1	42	.691	1	.754	1
61	-	-	+	+	+	+	-	30	.271	26273	240.000	3762	15	.126	4	.427	1
62	+	-	+	+	+	+	-	29	.312	63	1.206	7	38	.525	1	.560	1
63	-	+	+	+	+	+	-	22	.257	1551	21.853	186	16	.166	14	1.022	4
64	+	+	+	+	+	+	-	57	1.013	49	1.039	2	44	.825	22	1.851	3
65	-	-	-	-	-	-	+	13	.109	>25381	>240.000	>5428	18	.148	12	.637	4
66	+	-	-	-	-	-	+	21	.221	89	1.542	18	31	.321	2	.458	1
67	-	+	-	-	-	-	+	11	.109	204	2.580	50	13	.137	19	1.201	3
68	+	+	-	-	-	-	+	27	.328	1574	22.544	173	28	.404	39	2.220	5
69	-	-	+	-	-	-	+	19	.165	1132	11.718	180	11	.110	>879	>41.843	>160
70	+	-	+	-	-	-	+	21	.178	244	4.308	44	21	.207	3	.496	1
71	-	+	+	-	-	-	+	10	.105	372	4.522	72	12	.108	52	2.034	12
72	+	+	+	-	-	-	+	20	.263	989	15.634	126	34	.470	19	1.266	3
73	-	-	-	+	-	-	+	15	.130	1698	17.242	249	18	.162	>1152	>26.319	>21
74	+	-	-	+	-	-	+	24	.334	207	3.613	18	26	.340	12	1.040	2
75	-	+	-	+	-	-	+	21	.256	374	5.741	55	21	.300	>210	>13.274	>32
76	+	+	-	+	-	-	+	45	.891	>11793	>240.000	>905	47	.853	>1350	>57.321	>37
77	-	-	+	+	-	-	+	15	.157	1303	15.645	205	17	.162	>396	>19.299	>43
78	+	-	+	+	-	-	+	29	.371	419	7.022	41	35	.461	>374	>25.805	>51
79	-	+	+	+	-	-	+	16	.196	435	7.246	80	22	.280	>224	>14.966	>40
80	+	+	+	+	-	-	+	46	.820	>10822	>240.000	>935	45	.839	>294	>25.827	>24
81	-	-	-	-	+	-	+	9	.075	227	2.521	50	16	.131	39	1.346	5
82	+	-	-	-	+	-	+	21	.208	8	.200	2	19	.196	5	.476	1
83	-	+	-	-	+	-	+	12	.099	>23524	>240.000	>3947	12	.104	>263	>14.801	>50
84	+	+	-	-	+	-	+	23	.272	240	3.270	20	17	.183	7	.654	1

Problem Characteristics								Branch-Bound					Cutting Plane				
Problem No.	A	B	C	D	E	F	G	First LP	From LP to IP				First LP	From LP to IP			
								H	I	J	K	L	H	I	J	K	L
85	-	-	+	-	+	-	+	14	.123	>31530	>240.000	>3535	10	.074	>462	>46.081	>37
86	+	-	+	-	+	-	+	18	.151	20	.382	3	18	.171	3	.444	1
87	-	+	+	-	+	-	+	9	.066	133	1.651	35	18	.164	>269	>15.037	>43
88	+	+	+	-	+	-	+	22	.254	8087	129.142	963	37	.702	>197	>15.082	>20
89	-	-	-	+	+	-	+	21	.179	4870	61.836	876	13	.116	>1120	>22.297	>14
90	+	-	-	+	+	-	+	32	.494	4	.196	1	51	.684	43	2.818	6
91	-	+	-	+	+	-	+	18	.190	1797	23.802	204	20	.244	>281	>14.740	>25
92	+	+	-	+	+	-	+	45	.805	>11298	>240.000	>840	60	1.008	>354	>31.177	>41
93	-	-	+	+	+	-	+	19	.176	334	4.696	42	12	.094	>237	>9.806	>30
94	+	-	+	+	+	-	+	26	.336	54	.803	2	21	.228	13	.789	1
95	-	+	+	+	+	-	+	27	.331	3784	55.863	511	22	.248	>174	>9.372	>20
96	+	+	+	+	+	-	+	40	.646	1668	35.452	123	37	.617	>197	>15.170	>20
97	-	-	-	-	-	+	+	15	.103	>27728	>240.000	>5260	12	.087	11	.486	1
98	+	-	-	-	-	+	+	23	.249	77	1.329	9	27	.315	7	.696	1
99	-	+	-	-	-	+	+	16	.151	1	.134	1	11	.127	1	.503	1
100	+	+	-	-	-	+	+	28	.383	>20714	>240.000	>1343	26	.402	82	6.385	16
101	-	-	+	-	-	+	+	15	.128	16793	187.246	3371	10	.092	71	2.749	11
102	+	-	+	-	-	+	+	27	.260	229	3.352	33	24	.249	4	.470	1
103	-	+	+	-	-	+	+	17	.171	>25023	>240.000	>4647	12	.126	42	1.628	6
104	+	+	+	-	-	+	+	27	.311	1460	24.218	174	25	.308	12	.978	2
105	-	-	-	+	-	+	+	19	.217	>21890	>240.000	>3882	18	.179	>181	>9.908	>26
106	+	-	-	+	-	+	+	21	.265	399	6.813	38	47	.645	14	1.143	2
107	-	+	-	+	-	+	+	24	.332	2571	37.910	333	18	.217	>1054	>23.120	>9
108	+	+	-	+	-	+	+	69	1.312	>9982	>240.000	>858	47	.865	>571	>50.683	>61
109	-	-	+	+	-	+	+	25	.235	>23994	>240.000	>4223	29	.270	>249	>21.210	>27
110	+	-	+	+	-	+	+	43	.664	1116	19.420	91	37	.477	6	.758	1
111	-	+	+	+	-	+	+	23	.287	>17868	>240.000	>1813	24	.371	>382	>25.058	>78
112	+	+	+	+	-	+	+	40	.707	>11255	>240.000	> 990	52	.904	>320	>27.008	>33
113	-	-	-	-	+	+	+	16	.123	161	1.659	31	12	.103	20	.734	4

Problem Characteristics								Branch-Bound					Cutting Plane				
Problem No.	A	B	C	D	E	F	G	First LP		From LP to IP			First LP		From LP to IP		
								H	I	J	K	L	H	I	J	K	L
114	+	-	-	-	+	+	+	17	.157	1	.116	1	18	.182	4	.491	1
115	-	+	-	-	+	+	+	10	.105	34	.637	12	13	.140	32	1.412	8
116	+	+	-	-	+	+	+	37	.497	392	6.334	36	22	.267	23	1.578	5
117	-	-	+	-	+	+	+	14	.121	>30265	>240.000	>3577	8	.084	>211	>8.904	>23
118	+	-	+	-	+	+	+	24	.270	146	2.242	18	21	.159	10	.578	2
119	-	+	+	-	+	+	+	9	.083	248	3.954	72	15	.145	>223	>28.557	>37
120	+	+	+	-	+	+	+	27	.293	767	10.477	76	25	.330	18	1.366	2
121	-	-	-	+	+	+	+	12	.120	1655	21.015	210	22	.210	23	1.172	4
122	+	-	-	+	+	+	+	28	.370	440	8.757	52	23	.334	6	.734	1
123	-	+	-	+	+	+	+	20	.228	558	8.221	81	13	.176	22	1.343	5
124	+	+	-	+	+	+	+	42	.698	>10319	>240.000	>1000	53	.943	>163	>14.000	>23
125	-	-	+	+	+	+	+	22	.245	3500	44.429	536	14	.148	>273	>26.834	>26
126	+	-	+	+	+	+	+	25	.311	1	.165	1	35	.399	1	.588	1
127	-	+	+	+	+	+	+	19	.266	4736	78.322	887	19	.227	>1289	>42.415	>50
128	+	+	+	+	+	+	+	53	1.047	8729	179.946	663	39	.630	>189	>14.557	>18

APPENDIX C

REPLICATE II OF DATA COLLECTED

REPLICATE II OF DATA COLLECTED

Problem No.	Problem Characteristics							Branch-Bound					Cutting Plane				
								First LP		From LP to IP			First LP		From LP to IP		
	A	B	C	D	E	F	G	H	I	J	K	L	H	I	J	K	L
1	-	-	-	-	-	-	-	12	.070	1	.113	1	13	.111	3	.447	1
2	+	-	-	-	-	-	-	31	.282	775	13.087	107	22	.235	2	.496	1
3	-	+	-	-	-	-	-	13	.110	32	.527	6	12	.097	3	.514	1
4	+	+	-	-	-	-	-	33	.424	1	.169	1	23	.283	1	.541	1
5	-	-	+	-	-	-	-	15	.115	2	.180	2	13	.092	1	.354	1
6	+	-	+	-	-	-	-	23	.315	4	.444	4	20	.160	3	.494	1
7	-	+	+	-	-	-	-	12	.104	1	.166	1	18	.226	13	.801	1
8	+	+	+	-	-	-	-	22	.238	1	.201	1	28	.337	3	.641	1
9	-	-	-	+	-	-	-	13	.127	1	.138	1	12	.117	6	.507	1
10	+	-	-	+	-	-	-	29	.374	404	7.287	39	33	.463	1	.623	1
11	-	+	-	+	-	-	-	28	.341	403	6.143	59	16	.188	1	.476	1
12	+	+	-	+	-	-	-	46	.799	1731	37.201	139	41	.764	10	1.015	1
13	-	-	+	+	-	-	-	16	.130	1554	24.446	417	19	.174	2	.470	1
14	+	-	+	+	-	-	-	27	.529	1238	21.363	103	25	.299	1	.569	1
15	-	+	+	+	-	-	-	16	.183	4	.269	3	20	.229	3	.535	1
16	+	+	+	+	-	-	-	32	.551	50	1.441	5	44	.629	1	.742	1
17	-	-	-	-	+	-	-	13	.090	>28029	>240.000	>3180	9	.060	1	.363	1
18	+	-	-	-	+	-	-	20	.200	1	.212	1	26	.228	5	.458	1
19	-	+	-	-	+	-	-	15	.120	2	.166	1	14	.109	5	.476	2
20	+	+	-	-	+	-	-	28	.312	1	.141	1	21	.245	3	.513	1
21	-	-	+	-	+	-	-	9	.057	1	.095	1	14	.091	3	.327	1
22	+	-	+	-	+	-	-	18	.135	1	.135	1	19	.142	1	.408	1
23	-	+	+	-	+	-	-	9	.073	3	.157	2	13	.104	4	.435	1
24	+	+	+	-	+	-	-	23	.269	2	.188	1	16	.159	1	.487	1
25	-	-	-	+	+	-	-	18	.195	21122	240.000	4360	19	.147	4	.419	1
26	+	-	-	+	+	-	-	31	.367	91	1.873	9	34	.355	3	.609	1
27	-	+	-	+	+	-	-	19	.205	71	2.085	30	15	.187	9	.694	1

Problem Characteristics								Branch-Bound					Cutting Plane				
Problem No.	A	B	C	D	E	F	G	First LP		From LP to IP			First LP		From LP to IP		
								H	I	J	K	L	H	I	J	K	L
28	+	+	-	+	+	-	-	23	.427	5	.340	1	40	.605	14	1.285	2
29	-	-	+	+	+	-	-	20	.183	1	.127	1	18	.166	3	.442	1
30	+	-	+	+	+	-	-	19	.204	1	.149	1	27	.334	1	.631	1
31	-	+	+	+	+	-	-	21	.255	23	.567	5	23	.272	27	1.655	4
32	+	+	+	+	+	-	-	45	.883	10233	240.000	999	52	.937	19	1.498	2
33	-	-	-	-	-	+	-	11	.084	103	1.256	16	11	.069	8	.530	2
34	+	-	-	-	-	+	-	24	.272	112	2.292	20	20	.209	1	.400	1
35	-	+	-	-	-	+	-	16	.136	104	1.351	21	12	.113	1	.425	1
36	+	+	-	-	-	+	-	26	.340	1168	15.747	114	20	.321	4	.627	1
37	-	-	+	-	-	+	-	18	.177	21029	240.000	3255	12	.137	6	.569	1
38	+	-	+	-	-	+	-	26	.208	1	.142	1	22	.407	1	.205	1
39	-	+	+	-	-	+	-	15	.145	24362	240.000	3249	15	.175	16	.933	3
40	+	+	+	-	-	+	-	30	.336	1	.178	1	26	.351	18	1.371	6
41	-	-	-	+	-	+	-	13	.124	777	8.783	116	18	.152	19	1.013	4
42	+	-	-	+	-	+	-	33	.488	43	.867	3	32	.401	15	.989	2
43	-	+	-	+	-	+	-	15	.190	255	3.380	32	13	.298	6	.989	2
44	+	+	-	+	-	+	-	41	.759	722	16.502	53	39	.946	7	.932	1
45	-	-	+	+	-	+	-	16	.141	19	.341	4	13	.138	9	.717	3
46	+	-	+	+	-	+	-	35	.463	586	10.937	67	24	.315	1	.586	1
47	-	+	+	+	-	+	-	22	.250	5	.258	2	18	.213	93	4.709	26
48	+	+	+	+	-	+	-	49	.840	1328	31.760	127	32	.517	12	1.384	2
49	-	-	-	-	+	+	-	14	.129	14	.240	2	14	.067	12	.408	1
50	+	-	-	-	+	+	-	23	.214	1	.113	1	19	.196	1	.450	1
51	-	+	-	-	+	+	-	22	.167	782	8.914	158	18	.185	1	.413	1
52	+	+	-	-	+	+	-	21	.256	24	.454	2	28	.358	2	.568	1
53	-	-	+	-	+	+	-	10	.076	18	.311	5	15	.118	8	.536	4
54	+	-	+	-	+	+	-	25	.205	31	.597	6	18	.154	1	.406	1
55	-	+	+	-	+	+	-	10	.086	2	.139	2	13	.104	3	.427	3
56	+	+	+	-	+	+	-	28	.293	791	10.992	101	32	.426	9	.736	1

Problem Characteristics								Branch-Bound					Cutting Plane				
Problem No.	A	B	C	D	E	F	G	First LP		From LP to IP			First LP		From LP to IP		
								H	I	J	K	L	H	I	J	K	L
57	-	-	-	+	+	+	-	13	.089	329	3.950	54	20	.182	2	.368	1
58	+	-	-	+	+	+	-	30	.432	5	.426	4	21	.286	3	.709	1
59	-	+	-	+	+	+	-	19	.270	605	10.182	100	15	.148	14	1.095	3
60	+	+	-	+	+	+	-	32	.442	1	.164	1	70	1.173	1	.761	1
61	-	-	+	+	+	+	-	13	.103	>22052	>240.000	>3205	20	.178	1	.370	1
62	+	-	+	+	+	+	-	24	.285	39	.990	6	36	.439	1	.522	1
63	-	+	+	+	+	+	-	21	.271	1107	20.230	221	22	.256	19	1.146	3
64	+	+	+	+	+	+	-	46	.780	4	.224	1	51	1.063	8	1.103	1
65	-	-	-	-	-	-	+	11	.092	1916	24.124	620	13	.113	25	1.270	8
66	+	-	-	-	-	-	+	23	.233	64	1.215	11	21	.234	3	.449	1
67	-	+	-	-	-	-	+	16	.154	176	2.242	34	11	.118	24	.960	3
68	+	+	-	-	-	-	+	24	.290	50	.881	7	28	.346	14	.857	2
69	-	-	+	-	-	-	+	13	.109	218	2.525	48	11	.077	>376	>12.864	>67
70	+	-	+	-	-	-	+	23	.243	100	1.447	15	20	.204	5	.585	1
71	-	+	+	-	-	-	+	20	.224	284	3.983	50	14	.132	>353	>19.395	>41
72	+	+	+	-	-	-	+	27	.438	1730	30.212	182	28	.334	15	1.039	3
73	-	-	-	+	-	-	+	15	.157	>12664	>240.000	>5543	15	.137	>210	>8.633	>27
74	+	-	-	+	-	-	+	43	.562	4982	75.895	393	43	.578	5	.657	1
75	-	+	-	+	-	-	+	11	.117	2422	38.416	454	13	.147	89	4.084	8
76	+	+	-	+	-	-	+	42	.721	>11647	>240.000	>1008	45	.846	>491	>39.477	>42
77	-	-	+	+	-	-	+	15	.149	1355	20.156	257	18	.165	>171	>8.134	>23
78	+	-	+	+	-	-	+	25	.315	437	7.705	43	31	.409	14	.923	1
79	-	+	+	+	-	-	+	12	.133	1037	14.891	175	18	.211	>196	>11.312	>26
80	+	+	+	+	-	-	+	29	.501	>10459	>240.000	>1103	48	.842	>607	>59.619	>62
81	-	-	-	-	+	-	+	13	.106	>28136	>240.000	>2828	11	.100	>1241	>22.347	>41
82	+	-	-	-	+	-	+	15	.130	3	.156	2	29	.292	3	.457	1
83	-	+	-	-	+	-	+	15	.142	1125	13.697	268	10	.088	42	1.551	9
84	+	+	-	-	+	-	+	24	.325	>18072	>240.000	>1852	33	.424	16	1.180	7
85	-	-	+	-	+	-	+	16	.115	3877	43.584	974	12	.122	>194	>9.050	>32

Problem Characteristics								Branch-Bound					Cutting Plane				
Problem No.	A	B	C	D	E	F	G	First LP	From LP to IP				First LP	From LP to IP			
								H	I	J	K	L	H	I	J	K	L
86	+	-	+	-	+	-	+	15	.139	6	.228	2	19	.166	2	.399	1
87	-	+	+	-	+	-	+	13	.112	103	1.402	23	29	.323	3	.480	1
88	+	+	+	-	+	-	+	22	.277	121	1.632	12	22	.252	27	1.223	3
89	-	-	-	+	+	-	+	18	.161	78	1.070	12	14	.132	>377	>13.819	>49
90	+	-	-	+	+	-	+	28	.312	27	.452	2	24	.305	1	.500	1
91	-	+	-	+	+	-	+	13	.143	2854	36.260	408	20	.245	>191	>9.985	>19
92	+	+	-	+	+	-	+	44	.700	691	17.587	68	68	1.312	>394	>36.359	>39
93	-	-	+	+	+	-	+	12	.099	1590	17.427	283	17	.153	>638	>23.911	>77
94	+	-	+	+	+	-	+	33	.530	42	1.026	4	46	.661	28	1.245	2
95	-	+	+	+	+	-	+	28	.330	>19351	>240.000	>4236	21	.291	>312	>17.873	>44
96	+	+	+	+	+	-	+	50	.784	>10024	>240.000	>804	52	.864	>231	>17.280	>24
97	-	-	-	-	-	+	+	15	.138	665	7.934	175	11	.090	6	.478	2
98	+	-	-	-	-	+	+	26	.267	30	.873	10	21	.192	2	.440	1
99	-	+	-	-	-	+	+	11	.103	152	1.866	29	15	.124	12	.651	3
100	+	+	-	-	-	+	+	29	.375	>16617	>240.000	>1716	40	.495	15	.928	5
101	-	-	+	-	-	+	+	27	.264	229	3.447	33	13	.093	112	3.344	15
102	+	-	+	-	-	+	+	27	.292	>19837	>240.000	>1884	18	.176	2	.460	1
103	-	+	+	-	-	+	+	12	.118	4005	50.442	753	18	.213	29	1.772	7
104	+	+	+	-	-	+	+	23	.283	>17807	>240.000	>1716	16	.179	17	.999	6
105	-	-	-	+	-	+	+	19	.206	>23900	>240.00	>3889	16	.140	72	2.898	10
106	+	-	-	+	-	+	+	33	.443	953	15.422	83	40	.544	24	1.437	3
107	-	+	-	+	-	+	+	23	.266	1101	16.138	172	24	.325	19	1.182	3
108	+	+	-	+	-	+	+	44	.776	>11169	>240.000	>978	50	.900	>403	>33.051	>39
109	-	-	+	+	-	+	+	24	.198	>22279	>240.000	>3915	24	.226	91	3.435	22
110	+	-	+	+	-	+	+	32	.473	477	8.893	43	26	.349	21	1.272	2
111	-	+	+	+	-	+	+	23	.288	>17272	>240.000	>2566	18	.236	>213	>11.353	>27
112	+	+	+	+	-	+	+	41	.809	>11057	>240.000	>1052	43	.761	>572	>45.694	>50
113	-	-	-	-	+	+	+	13	.091	431	4.864	112	17	.151	6	.544	2
114	+	-	-	-	+	+	+	26	.255	26	.361	3	20	.200	3	.438	1

Problem Characteristics								Branch-Bound					Cutting Plane				
Problem No.	A	B	C	D	E	F	G	First LP		From LP to IP			First LP		From LP to IP		
								H	I	J	K	L	H	I	J	K	L
115	-	+	-	-	+	+	+	19	.193	3	.142	1	19	.219	22	1.245	5
116	+	+	-	-	+	+	+	21	.230	561	7.924	71	23	.236	23	1.301	6
117	-	-	+	-	+	+	+	12	.090	108	1.102	17	11	.077	18	.734	4
118	+	-	+	-	+	+	+	22	.253	2	.142	1	17	.190	3	.499	1
119	-	+	+	-	+	+	+	21	.186	216	2.598	43	9	.068	19	.843	8
120	+	+	+	-	+	+	+	25	.399	>16634	>240.000	>1773	25	.273	39	1.833	6
121	-	-	-	+	+	+	+	17	.156	298	4.044	75	21	.165	>483	>20.063	>58
122	+	-	-	+	+	+	+	45	.592	5	.295	1	28	.284	9	.729	1
123	-	+	-	+	+	+	+	19	.229	1081	16.120	135	18	.183	49	2.428	8
124	+	+	-	+	+	+	+	43	.719	1375	28.643	75	69	1.308	>1336	>60.190	>34
125	-	-	+	+	+	+	+	12	.103	1148	16.550	311	14	.125	>273	>25.265	>26
126	+	-	+	+	+	+	+	25	.326	33	.579	3	37	.486	3	.570	1
127	-	+	+	+	+	+	+	21	.250	1327	20.772	214	23	.231	>235	>11.840	>25
128	+	+	+	+	+	+	+	59	1.283	5665	132.886	414	60	1.118	>406	>34.055	>47

APPENDIX D

ANALYSIS OF VARIANCE TABLE FOR THE CUTTING PLANE RESULTS

BMD02V - ANALYSIS OF VARIANCE FOR FACTORIAL DESIGN
HEALTH SCIENCES COMPUTING FACILITY, UCLA

PROBLEM NO. 01

NUMBER OF VARIABLES 8
NUMBER OF REPLICATES 1

VARIABLE	NO. OF LEVELS
1	2
2	2
3	2
4	2
5	2
6	2
7	2
8	2

GRAND MEAN .96778

SOURCE OF VARIATION	DEGREES OF FREEDOM	SUMS OF SQUARES	MEAN SQUARES
1	1	.53888	.53888
2	1	25.31080	25.31080
3	1	.17724	.17724
4	1	.01599	.01599
5	1	12.24069	12.24069
6	1	.85368	.85368
7	1	6.24365	6.24365
8	1	.99454	.99454
12	1	.13825	.13825
13	1	.01475	.01475
14	1	.01386	.01386
15	1	.00362	.00362
16	1	.01508	.01508
17	1	.00553	.00553
18	1	.01258	.01258
23	1	.86511	.86511
24	1	.09285	.09285
25	1	1.08711	1.08711
26	1	.45515	.45515
27	1	.18502	.18502
28	1	3.63337	3.63337
34	1	.21881	.21881
35	1	.00425	.00425
36	1	.04993	.04993
37	1	.16131	.16131
38	1	.13929	.13929

45	1	.11915	.11915
46	1	.00092	.00092
47	1	.03396	.03396
48	1	.03161	.03161
56	1	.11846	.11846
57	1	.33648	.33648
58	1	.11002	.11002
67	1	.00002	.00002
68	1	.38864	.38864
78	1	2.78162	2.78162
123	1	.00090	.00090
124	1	.00162	.00162
125	1	.04371	.04371
126	1	.02773	.02773
127	1	.09020	.09020
128	1	.05598	.05598
134	1	.04395	.04395
135	1	.14243	.14243
136	1	.00542	.00542
137	1	.00271	.00271
138	1	.00409	.00409
145	1	.34576	.34576
146	1	.27091	.27091
147	1	.01402	.01402
148	1	.10007	.10007
156	1	.08773	.08773
157	1	.04901	.04901
158	1	.00267	.00267
167	1	.28779	.28779
168	1	.00084	.00084
178	1	.05836	.05836
234	1	.09272	.09272
235	1	.00137	.00137
236	1	.01329	.01329
237	1	.00022	.00022
238	1	.51825	.51825
245	1	.10656	.10656
246	1	.05464	.05464
247	1	.02968	.02968
248	1	.17007	.17007
256	1	.03861	.03861
257	1	.02659	.02659
258	1	.00007	.00007
267	1	.05945	.05945
268	1	.45512	.45512
278	1	2.57960	2.57960
345	1	.29451	.29451
346	1	.07094	.07094
347	1	.00925	.00925
348	1	.03837	.03837
356	1	.01066	.01066
357	1	.00035	.00035
358	1	.03926	.03926
367	1	.04672	.04672
368	1	.34659	.34659
378	1	.00005	.00005
456	1	.08020	.08020

457	1	.06776	.06776
458	1	.00010	.00010
467	1	.01465	.01465
468	1	.06369	.06369
478	1	.00506	.00506
567	1	.00332	.00332
568	1	.14417	.14417
578	1	.07005	.07005
678	1	.04232	.04232
1234	1	.00290	.00290
1235	1	.00425	.00425
1236	1	.01153	.01153
1237	1	.01485	.01485
1238	1	.14927	.14927
1245	1	.16622	.16622
1246	1	.10533	.10533
1247	1	.05838	.05838
1248	1	.00046	.00046
1256	1	.01737	.01737
1257	1	.05264	.05264
1258	1	.00005	.00005
1267	1	.08144	.08144
1268	1	.27009	.27009
1278	1	.02700	.02700
1345	1	.00036	.00036
1346	1	.03661	.03661
1347	1	.01641	.01641
1348	1	.00121	.00121
1356	1	.07022	.07022
1357	1	.19984	.19984
1358	1	.01746	.01746
1367	1	.05110	.05110
1368	1	.04587	.04587
1378	1	.01625	.01625
1456	1	.07750	.07750
1457	1	.10775	.10775
1458	1	.28169	.28169
1467	1	.34713	.34713
1468	1	.16808	.16808
1478	1	.01125	.01125
1567	1	.04579	.04579
1568	1	.12701	.12701
1578	1	.03700	.03700
1678	1	.08420	.08420
2345	1	.14298	.14298
2346	1	.12986	.12986
2347	1	.03516	.03516
2348	1	.01946	.01946
2356	1	.00260	.00260
2357	1	.00947	.00947
2358	1	.14455	.14455
2367	1	.00118	.00118
2368	1	.12399	.12399
2378	1	.02594	.02594
2456	1	.00447	.00447
2457	1	.05600	.05600
2458	1	.01114	.01114

2467	1	.01050	.01050
2468	1	.04306	.04306
2478	1	.07909	.07909
2567	1	.20220	.20220
2568	1	.02028	.02028
2578	1	.17236	.17236
2678	1	.00036	.00036
3456	1	.01043	.01043
3457	1	.00468	.00468
3458	1	.00026	.00026
3467	1	.00491	.00491
3468	1	.02378	.02378
3478	1	.00744	.00744
3567	1	.04755	.04755
3568	1	.01093	.01093
3578	1	.00266	.00266
3678	1	.00288	.00288
4567	1	.01575	.01575
4568	1	.26772	.26772
4578	1	.00536	.00536
4678	1	.07474	.07474
5678	1	.11082	.11082
12345	1	.00476	.00476
12346	1	.02515	.02515
12347	1	.09160	.09160
12348	1	.04550	.04550
12356	1	.00015	.00015
12357	1	.10200	.10200
12358	1	.06920	.06920
12367	1	.01258	.01258
12368	1	.03359	.03359
12378	1	.12059	.12059
12456	1	.13730	.13730
12457	1	.01548	.01548
12458	1	.28615	.28615
12467	1	.00019	.00019
12468	1	.04613	.04613
12478	1	.13589	.13589
12567	1	.02154	.02154
12568	1	.00262	.00262
12578	1	.10317	.10317
12678	1	.02516	.02516
13456	1	.09706	.09706
13457	1	.03207	.03207
13458	1	.05244	.05244
13467	1	.03454	.03454
13468	1	.03647	.03647
13478	1	.01975	.01975
13567	1	.00794	.00794
13568	1	.00227	.00227
13578	1	.04948	.04948
13678	1	.06336	.06336
14567	1	.00008	.00008
14568	1	.01881	.01881
14578	1	.02040	.02040
14678	1	.00028	.00028
15678	1	.05928	.05928

23456	1	.03810	.03810
23457	1	.02169	.02169
23458	1	.00335	.00335
23467	1	.09654	.09654
23468	1	.02019	.02019
23478	1	.00166	.00166
23567	1	.02553	.02553
23568	1	.00191	.00191
23578	1	.33184	.33184
23678	1	.06041	.06041
24567	1	.05401	.05401
24568	1	.25335	.25335
24578	1	.06206	.06206
24678	1	.07008	.07008
25678	1	.19419	.19419
34567	1	.04381	.04381
34568	1	.00034	.00034
34578	1	.07838	.07838
34678	1	.08420	.08420
35678	1	.05724	.05724
45678	1	.00092	.00092
123456	1	.06272	.06272
123457	1	.04687	.04687
123458	1	.13247	.13247
123467	1	.52783	.52783
123468	1	.01277	.01277
123478	1	.00565	.00565
123567	1	.00651	.00651
123568	1	.00112	.00112
123578	1	.05085	.05085
123678	1	.00192	.00192
124567	1	.00014	.00014
124568	1	.04088	.04088
124578	1	.09200	.09200
124678	1	.03684	.03684
125678	1	.02217	.02217
134567	1	.06301	.06301
134568	1	.00074	.00074
134578	1	.14524	.14524
134678	1	.05168	.05168
135678	1	.13017	.13017
145678	1	.00008	.00008
234567	1	.04547	.04547
234568	1	.01639	.01639
234578	1	.04066	.04066
234678	1	.01302	.01302
235678	1	.02987	.02987
245678	1	.00426	.00426
345678	1	.00849	.00849
1234567	1	.05951	.05951
1234568	1	.01609	.01609
1234578	1	.04407	.04407
1234678	1	.00011	.00011
1235678	1	.02055	.02055
1245678	1	.00284	.00284
1345678	1	.04909	.04909
2345678	1	.00006	.00006
RESIDUAL	1	.00012	.00012
TOTAL	255	74.10647	

APPENDIX E

ANALYSIS OF VARIANCE TABLE FOR BRANCH AND BOUND RESULTS

BMD02V - ANALYSIS OF VARIANCE FOR FACTORIAL DESIGN
HEALTH SCIENCES COMPUTING FACILITY, UCLA

PROBLEM NO. 01

NUMBER OF VARIABLES 8
NUMBER OF REPLICATES 1

VARIABLE	NO. OF LEVELS
1	2
2	2
3	2
4	2
5	2
6	2
7	2
8	2

GRAND MEAN .87565

SOURCE OF VARIATION	DEGREES OF FREEDOM	SUMS OF SQUARES	MEAN SQUARES
1	1	.00059	.00059
2	1	2.98627	2.98627
3	1	.28494	.28494
4	1	.93132	.93132
5	1	1.12245	1.12245
6	1	.05402	.05402
7	1	.46171	.46171
8	1	.16896	.16896
12	1	.01097	.01097
13	1	.02323	.02323
14	1	.01510	.01510
15	1	.00379	.00379
16	1	.03257	.03257
17	1	.00565	.00565
18	1	.01942	.01942
23	1	.06596	.06596
24	1	.00417	.00417
25	1	.00235	.00235
26	1	.02073	.02073
27	1	.16418	.16418
28	1	.00003	.00003
34	1	.18469	.18469
35	1	.00419	.00419
36	1	.19363	.19363
37	1	.00398	.00398
38	1	.03645	.03645

45	1	.01375	.01375
46	1	.00000	.00000
47	1	.04726	.04726
48	1	.20012	.20012
56	1	.00570	.00570
57	1	.02583	.02583
58	1	.00786	.00786
67	1	.02518	.02518
68	1	.00655	.00655
78	1	1.19056	1.19056
123	1	.00009	.00009
124	1	.07193	.07193
125	1	.01083	.01083
126	1	.02472	.02472
127	1	.04753	.04753
128	1	.00584	.00584
134	1	.00936	.00936
135	1	.17640	.17640
136	1	.00627	.00627
137	1	.09976	.09976
138	1	.01289	.01289
145	1	.00087	.00087
146	1	.00009	.00009
147	1	.00283	.00283
148	1	.02119	.02119
156	1	.00145	.00145
157	1	.00001	.00001
158	1	.00626	.00626
167	1	.00446	.00446
168	1	.08724	.08724
178	1	.01407	.01407
234	1	.04171	.04171
235	1	.00690	.00690
236	1	.00000	.00000
237	1	.00088	.00088
238	1	.14031	.14031
245	1	.02070	.02070
246	1	.00017	.00017
247	1	.08124	.08124
248	1	.00034	.00034
256	1	.05245	.05245
257	1	.08941	.08941
258	1	.00186	.00186
267	1	.00106	.00106
268	1	.00005	.00005
278	1	.41334	.41334
345	1	.02100	.02100
346	1	.00945	.00945
347	1	.03408	.03408
348	1	.10282	.10282
356	1	.00189	.00189
357	1	.04421	.04421
358	1	.17551	.17551
367	1	.01768	.01768
368	1	.12872	.12872
378	1	.05406	.05406
456	1	.01815	.01815

457	1	.00056	.00056
458	1	.10487	.10487
467	1	.01167	.01167
468	1	.02568	.02568
478	1	.00824	.00824
567	1	.00506	.00506
568	1	.00149	.00149
578	1	.00045	.00045
678	1	.00367	.00367
1234	1	.00188	.00188
1235	1	.02705	.02705
1236	1	.00053	.00053
1237	1	.01849	.01849
1238	1	.01654	.01654
1245	1	.04433	.04433
1246	1	.00011	.00011
1247	1	.00036	.00036
1248	1	.00990	.00990
1256	1	.04779	.04779
1257	1	.01744	.01744
1258	1	.04034	.04034
1267	1	.01591	.01591
1268	1	.00661	.00661
1278	1	.00046	.00046
1345	1	.04459	.04459
1346	1	.00028	.00028
1347	1	.00228	.00228
1348	1	.01194	.01194
1356	1	.04811	.04811
1357	1	.04877	.04877
1358	1	.00190	.00190
1367	1	.00843	.00843
1368	1	.00923	.00923
1378	1	.02873	.02873
1456	1	.09138	.09138
1457	1	.01595	.01595
1458	1	.00006	.00006
1467	1	.14883	.14883
1468	1	.06594	.06594
1478	1	.07067	.07067
1567	1	.01634	.01634
1568	1	.00077	.00077
1578	1	.00001	.00001
1678	1	.00050	.00050
2345	1	.00096	.00096
2346	1	.05164	.05164
2347	1	.00058	.00058
2348	1	.00003	.00003
2356	1	.08090	.08090
2357	1	.00317	.00317
2358	1	.00875	.00875
2367	1	.10601	.10601
2368	1	.00211	.00211
2378	1	.00679	.00679
2456	1	.00107	.00107
2457	1	.03703	.03703
2458	1	.14274	.14274

2467	1	.03759	.03759
2468	1	.05112	.05112
2478	1	.00120	.00120
2567	1	.00383	.00383
2568	1	.04695	.04695
2578	1	.01483	.01483
2678	1	.07867	.07867
3456	1	.01551	.01551
3457	1	.05222	.05222
3458	1	.00533	.00533
3467	1	.00030	.00030
3468	1	.03253	.03253
3478	1	.04069	.04069
3567	1	.02811	.02811
3568	1	.00898	.00898
3578	1	.02693	.02693
3678	1	.02493	.02493
4567	1	.07786	.07786
4568	1	.03963	.03963
4578	1	.05252	.05252
4678	1	.04712	.04712
5678	1	.04625	.04625
12345	1	.00324	.00324
12346	1	.00276	.00276
12347	1	.00036	.00036
12348	1	.00024	.00024
12356	1	.03284	.03284
12357	1	.00240	.00240
12358	1	.03514	.03514
12367	1	.00001	.00001
12368	1	.04125	.04125
12378	1	.02302	.02302
12456	1	.07232	.07232
12457	1	.00131	.00131
12458	1	.00445	.00445
12467	1	.05272	.05272
12468	1	.12636	.12636
12478	1	.00886	.00886
12567	1	.00110	.00110
12568	1	.01296	.01296
12578	1	.00567	.00567
12678	1	.00002	.00002
13456	1	.00049	.00049
13457	1	.00130	.00130
13458	1	.12368	.12368
13467	1	.04451	.04451
13468	1	.02003	.02003
13478	1	.01996	.01996
13567	1	.04670	.04670
13568	1	.00000	.00000
13578	1	.02518	.02518
13678	1	.00309	.00309
14567	1	.01318	.01318
14568	1	.00691	.00691
14578	1	.05562	.05562
14678	1	.01596	.01596
15678	1	.00054	.00054

23456	1	.02942	.02942
23457	1	.08060	.08060
23458	1	.05016	.05016
23467	1	.06738	.06738
23468	1	.07602	.07602
23478	1	.03702	.03702
23567	1	.06258	.06258
23568	1	.00224	.00224
23578	1	.03003	.03003
23678	1	.00849	.00849
24567	1	.01024	.01024
24568	1	.02802	.02802
24578	1	.00420	.00420
24678	1	.05468	.05468
25678	1	.00603	.00603
34567	1	.09559	.09559
34568	1	.09795	.09795
34578	1	.03592	.03592
34678	1	.06519	.06519
35678	1	.00000	.00000
45678	1	.01567	.01567
123456	1	.04630	.04630
123457	1	.01110	.01110
123458	1	.10673	.10673
123467	1	.19110	.19110
123468	1	.00030	.00030
123478	1	.02569	.02569
123567	1	.00413	.00413
123568	1	.01825	.01825
123578	1	.00000	.00000
123678	1	.00009	.00009
124567	1	.05220	.05220
124568	1	.11325	.11325
124578	1	.01921	.01921
124678	1	.00780	.00780
125678	1	.03026	.03026
134567	1	.05716	.05716
134568	1	.02382	.02382
134578	1	.01627	.01627
134678	1	.06822	.06822
135678	1	.02287	.02287
145678	1	.07699	.07699
234567	1	.00670	.00670
234568	1	.06119	.06119
234578	1	.00579	.00579
234678	1	.00087	.00087
235678	1	.09257	.09257
245678	1	.03177	.03177
345678	1	.00233	.00233
1234567	1	.01401	.01401
1234568	1	.11900	.11900
1234578	1	.02376	.02376
1234678	1	.02501	.02501
1235678	1	.01586	.01586
1245678	1	.01123	.01123
1345678	1	.07094	.07094
2345678	1	.00362	.00362
RESIDUAL	1	.00326	.00326
TOTAL	255	15.62658	

BIBLIOGRAPHY

1. Balas, E., "A Note on the Group Theoretic Approach to Integer Programming and the 0-1 Case," Operations Research, 21, 321-322, 1973.

2. Bartlett, M. S., "The Use of Transformations," Biometrics, 3, 39-52, 1947.

3. Bartlett, M. S. and D. G. Kendal, "The statistical Analysis of Variance-Heterogeneity and the Logarithmic Transformation," Supplement to the Journal of the Royal Statistical Society, B, 1, 128-138, 1946.

4. Beale, E. M. L. and J. A. Tomlin, "An Integer Programming Approach to a Class of Combinatorial Problems," Mathematical Programming, 3, 339-344, 1972.

5. Box, G. E. P., "Non-normality and Tests on Variances," Biometrika, 40, 318-335, 1953.

6. Box, G. E. P., "Some Theorems on Quadratic Forms Applied in the Study of ANOVA Problems: I. Effect of Inequality of Variance in the One-Way Classification," Annual Mathematical Statistics, 25, 290-302, 1954.

7. Box, G. E. P. and D. R. Cox, "An Analysis of Transformations," Journal of Royal Statistical Society, B, 26, 211-252, 1964.

8. Box, G. E. P. and J. S. Hunter, "The 2^{k-p} Fractional Factorial Designs," Technometrics, 3, 311-352, 1961.

9. Brown, M. B. and A. B. Forsythe, "Robust Tests for the Equality of Variances," Journal of the American Statistical Association, 69, 364-367, 1974.

10. Charnes, A., W. M. Raike, J. D. Stutz, and A. S. Walters, "On Generation of Test Problems for Linear Programming Codes," Communications of the ACM, 17, 583-586, 1974.

11. Cochran, W. G., "Some Consequences When the Assumptions for the Analysis of Variance Are Not Satisfied," Biometrics, 3, 22-38, 1947.

12. Colville, A. R., "A Comparative Study of Nonlinear Programming Codes," Proceedings of the Princeton Symposium on Mathematical Programming, H. W. Kuhn, (Ed.), Princeton University Press, 487-501, 1970.

13. Dantzig, G. R., Linear Programming and Extensions, Princeton University Press, 1963.

14. Davies, O. L., Design and Analysis of Industrial Experiments, Hafner Publishing Company (New York), 1954.

15. Dolby, J. L., "A Quick Method for Choosing a Transformation," Technometrics, 5, 317-325, 1963.

16. Draper, N. R. and W. G. Hunter, "Transformations: Some Examples Revisited," Technometrics, 11, 23-40, 1969.

17. Eisenhart, C., "The Assumptions Underlying the Analysis of Variance," Biometrics, 3, 1-21, 1947.

18. Geoffrion, A. M. and G. W. Graves, "Multicommodity Distribution System Design by Benders Decomposition," Management Science, 20, 822-844, 1974.

19. Geoffrion, A. M. and R. E. Marsten, "Integer Programming: A Framework and State-of-the-Art Survey," Management Science, 18, 465-491, 1972.

20. Geoffrion, A. M., "Recent Computational Experience with Three Classes of Integer Linear Programs," Working Paper P-3699, The RAND Corporation, Santa Monica, Calif., 1967.

21. Giglio, R. J. and H. M. Wagner, "Approximate Solutions to the Three Machine Scheduling Problem," Operations Research, 12, 305-324, 1964.

22. Glover, G., D. Karney, D. Klingman, and A. Napier, "A Computation Study on Start Procedures, Basis Change Criteria, and Solution Algorithms for Transportation Problems," Management Science, 20, 793-813, 1974.

23. Gomory, R. E., "On the Relation Between Integer and Non-Integer Solutions to Linear Programs," Proceedings, National Academy of Science, 53, 260-265, 1965.

24. Gorry, G. A. and J. F. Shapiro, "An Adaptive Group Theoretic Algorithm for Integer Programming Problems," Management Science, 17, 285-306, 1971.

25. Greenberg, H. and R. L. Hegerich, "A Branch Search Algorithm for the Knapsack Problem," Management Science, 16, 327-332, 1970.

26. Haldi, J., "Fixed Charge Problem and Integer Programming," Working Paper No. 13, Graduate School of Business, Stanford University, 1964.

27. Haldi, J., "25 Integer Linear Programming Test Problems," Working Paper No. 43, Graduate School of Business, Stanford University, 1964.

28. Haldi, J. and L. M. Isaacson, "A Computer Code for Integer Solutions to Linear Programs," Operations Research, 13, 946-959, 1965.

29. Himmelblau, D. M., Applied Nonlinear Programming, McGraw-Hill (New York), 1972.

30. Jeroslow, R. G., "Comments on Integer Hulls of Two Linear Constraints," Operations Research, 19, 161-169, 1971.

31. Jeroslow, R. G., "The Number of Faces in the Integer Hull of Two Dimensional Asymptotic Programs as a Function of the Determinant," Management Sciences Research Report No. 221, Carnegie-Mellon University, 1970.

32. Jeroslow, R. G. and K. O. Kortanek, "On an Algorithm of Gomory," SIAM Applied Mathematics, 21, 55-60, 1971.

33. Karp, R. M., "Reducibility Among Combinatorial Problems," Technical Report 3, Computer Science, University of Calif., 1972.

34. Khumawala, B. M., "An Efficient Branch and Bound Algorithm for the Warehouse Location Problem," Management Science, 18, 718-731, 1972.

35. Klingman, D., A. Napier, and J. Stutz, "NETGEN, A Program for Generating Large Scale Capacitated Assignment Transportation and Minimum Cost Flow Network Problems," Management Science, 20, 814-821, 1974.

36. Kuehn, A. A. and M. J. Hamburger, "A Heuristic Program for Locating Warehouse," Management Science, 9, 643-666, 1963.

37. Land, A. H. and A. G. Doig, "An Automatic Method for Solving Discrete Programming Problems," Econometrica, 28, 497-520, 1960.

38. Land, A. H. and S. Power, Fortran Codes for Mathematical Programming Linear, Quadratic, and Discrete, John Wiley & Sons (New York), 1973.

39. Levene, H., "Robust Tests for Equality of Variances," in I. Olkin (Ed.), Contributions to Probability and Statistics, Stanford University Press, 278-292, 1960.

40. McGinnis, L. F., Approximate and Exact Solution Procedures for a Class of Facilities Location Problems, Ph.D. Dissertation, North Carolina State University, Raleigh, North Carolina, 1975.

41. Michaels, W. M. and R. P. O'Neill, "A Mathematical Program Generator," Presented at the ORSA/TIMS Joint National Meeting, Chicago, Ill., April 30, 1975.

42. Nelson, W., "Analysis of Residuals from Censored Data," Technometrics, 15, 697-715, 1973.

43. Nelson, G. J. and W. Nelson, "A Comparison of Methods for Analyzing Censored Life Data to Estimate Relationships Between Stress and Product Life," IEEE Transactions on Reliability, 23, 2-10, 1974.

44. Nemhauser, G. L. and Z. Ullman, "Discrete Dynamic Programming and Capital Allocation," Management Science, 15, 494-505, 1969.

45. Pierce, J. F., "Application of Combinatorial Programming to a Class of All-Zero-One Integer Programming Problems," Management Science, 15, 191-209, 1968.

46. Rardin, R. L., Group Theoretic and Related Approaches to Fixed Charge Problems, Ph.D. dissertation, School of Industrial and Systems Engineering, Georgia Institute of Technology, 1974.

47. Rardin, R. L. and V. E. Unger, "Solving Fixed Charge Network Problems with Group Theory-Based Penalties," forthcoming in Naval Research Logistics Quarterly.

48. Sà, G., "Branch-and-Bound and Approximate Solutions to the Capacitated Plant-Location Problem," Operations Research, 17, 1005-1016, 1969.

49. Sampford, M. R. and J. Taylor, "Censored Observations in Randomized Block Experiments," Journal of Royal Statistical Society, B, 21, 214-237, 1959.

50. Schmeiser, B. W., "Some Practical Reasons for Using Nonlinear Techniques in Certain Linear Programming Applications," School of Industrial and Systems Engineering, Georgia Institute of Technology, presented at the ORSA/TIMS National Meeting, Chicago, Illinois, 1975.

51. Spielberg, K., "Algorithms for the Simple Plant-Location Problem with Some Side Conditions," Operations Research, 17, 85-111, 1969.

52. Taylor, J., "The Analysis of Designed Experiments with Censored Observations," Biometrics, 29, 35-43, 1973.

53. Thompson, G. L., "The Stopped Simplex Method: 1. Basic Theory for Mixed Integer Programming," Revue Franciase De Recherche Operationnelle, 5, 159-178, 1964.

54. Trauth, C. A. and R. E. Woolsey, "Integer Linear Programming: A Study in Computational Efficiency," Management Science, 15, 481-493, 1969.

55. Trotter, L. E., Jr. and C. M. Shetty, "An Algorithm for the Bounded Variable Integer Programming Problem," School of Industrial and Systems Engineering, Georgia Institute of Technology, 1971.

56. Scheffe, H., The Analysis of Variance, John Wiley (New York), 1959.

57. Spielberg, K., "Plant Location with Generalized Search Origin," Management Science, 16, 165-178, 1969.

58. Wahi, P. N. and G. H. Bradley, "Integer Programming Test Problems," Technical Report, No. 28, Yale University, 1969.

59. Williams, H. P., "Experiments in the Formulation of Integer Programming Problems," forthcoming in Mathematical Programming.

VITA

Benjamin Wei-Yuh Lin was born in Taipei, Taiwan on July 18, 1947. He was graduated from Chien-Kou High School, Taipei, Taiwan, in 1965.

He received the degree of Bachelor in Mechanical Engineering from National Taiwan University, Taipei, Taiwan, in 1969. After serving one year in the Nationalist Chinese Army, he came to the United States for graduate studies. He was awarded a Master of Science in Applied Mathematics and Statistics from the State University of New York at Stony Brook in August, 1971.

Mr. Lin entered the doctoral program in the School of Industrial and Systems Engineering of Georgia Institute of Technology in September, 1971. During his doctoral studies, he worked part-time with the Engineering Experiment Station of Georgia Institute of Technology. There he provided technical and management assistance to small businesses in the metropolitan Atlanta area. He was awarded the Ph.D. degree in 1975, and immediately accepted a position as an Assistant Professor in the Department of Industrial Engineering and Management Systems, Florida Technological University.

Mr. Lin was married in April, 1973 to Miss Josefina Shan-Ju Liu.